Locally Weighted Krill Herd Support Vector Regression for forecasting and trading Exchange Traded Funds

by

Georgios Sermpinis*

Charalampos Stasinakis**

Arman Hassannia Kalager***

May 2016

Abstract

This study introduces a Locally weighted Krill Herd - Support Vector Regression (KH-LSVR) model. The Krill Herd (KH) algorithm is a novel metaheuristic optimization technique inspired by the behaviour of krill herds. In KH-LSVR, the KH optimizes the Locally weighted Support Vector Regression (LSVR) parameters by balancing the search between local and global optima. The proposed model is applied to the task of forecasting and trading seven ETF stocks on a daily basis over the period 2006-2012. The KH-LSVR's efficiency is evaluated through three different fitness functions, while its statistical and trading performance is benchmarked seven traditional SVR structures. The inputs of the SVR models are selected through a novel statistical technique that involves three hundred forty eight linear and non-linear predictors, and the Model Confidence Set (MCS) test proposed by Hansen *et al.* (2011). The trading application is designed in order to validate the robustness of the algorithm under study and to provide empirical evidence in favour or against the Adaptive Market Hypothesis (AMH). In terms of the results, the KH-LSVR outperforms its counterparts in terms of statistical accuracy and trading efficiency, while the time varying trading performance of the models under study validates the AMH theory.

Keywords

Support Vector Regression, Krill Herd, ETF, Adaptive Market Hypothesis, Optimization

* University of Glasgow Business School, University of Glasgow, Gilbert Scott Building, Glasgow G12 8QQ, United Kingdom (Georgios.Sermpinis@glasgow.ac.uk)
**University of Glasgow Business School, University of Glasgow, Gilbert Scott Building, Glasgow G12 8QQ, United Kingdom (Charalampos.Stasinakis@glasgow.ac.uk)
*** Research Associate, University of Glasgow Business School, University of Glasgow, Gilbert Scott Building, Glasgow G12 8QQ, United Kingdom (a.hassanniakalager.1@research.gla.ac.uk)

1. Introduction

In the competitive financial world, identifying optimal solutions for a given financial problem is very important, but it can also prove utterly challenging. The financial task might not be well defined or might suffer from lack of necessary data. Even if this is not the case, simple constraints can often impede closed form solutions or the application of standard numerical methods (Lo, 2000; Gilli *et al.*, 2008). This usually can be overcome by restating the problem through relaxed assumptions that simplify it and consequently its solution. Logically, it should be preferable to adapt an optimization approach around the original problem. This can be achieved by heuristic optimization techniques (Hall and Posner, 2007). Their success in financial applications is well documented (Chang *et al.*, 2000; Gilli *et al.*, 2008; Oreski and Oreski, 2014; Aguilar-Rivera *et al.*, 2015).

Nonetheless, their use can be computationally demanding compared to traditional models while they can trap in local optima and suffer from over-fitting. Metaheuristics are problem-dependent techniques that can overcome these issues to some extent (Turmon 1998; Bernstein *et al.*, 2015). They achieve a trade-off between intensification (local search) and randomization (global search) by intelligent selection of random variables, without being problem dependent (Talbi, 2009). Martí *et al.* (2013) compare several heuristics and metaheuristic models and suggest that the best solutions are provided with metaheuristic approaches, while computational time is also decreased. This feature made them a popular approach for solving complex optimization problems. Their modeling is inspired by activities appearing in nature. For example, many algorithms evolve around species evolution and movement (Goldberg, 1989; Yang and Gandomi, 2012; Li *et al.*, 2014) while others are based on swarm behavior and intelligence (Liang *et al.*, 2006; Karaboga and Basturk, 2008; Yang, 2010; Popescu and Crama (2015)).

A novel bio-inspired metaheuristic method is the Krill Herd algorithm (KH) as proposed by Gandomi and Alavi (2012). KH simulates the herding behavior of krill individuals. Its objective functions are the minimum distances of krill from the food location and the location of the highest density of the herd. Each krill position is a time-dependent function formulated by three motions. The movement induced by other individuals, the foraging motion and a random physical diffusion. In KH the derivative information is not necessary because it uses a stochastic random search rather than a gradient search. Additionally, KH requires the fine tuning of only one parameter, in contrary to other metaheuristics algorithms (such as the particle swarm optimization and the harmony search). Gandomi and Alavi (2012) and Wang *et al.* (2014) compare the efficiency of KH against the most popular metaheuristics optimization models. In both studies, the KH present a superior performance. It is worth noting that there is no application of the KH in a financial forecasting framework.

Support Vector Regressions (SVRs) are non-linear data-adaptive regression techniques. SVRs main advantage is their ability to generate nonlinear decision boundaries through linear classifiers, while having a simple geometric interpretation (Suykens, 2002). SVR applications exhibit promising results in financial forecasting tasks (see amongst others Trafalis and Ince (2000), Hsu *et al.* (2009), Yeh *et al.* (2011) and Sermpinis *et al.* (2015)). Their main drawback, though, is the sensitivity to their parametrization process. For that reason, evolutionary techniques, especially GAs, are commonly combined with SVR in order to form superior forecasting hybrid structures. For example, Pai *et al.* (2006) apply epsilon SVR with genetically optimized parameters (GA-εSVR) in forecasting exchange rates. Huang *et al.* (2010) forecast the EUR/USD, GBP/USD, NZD/USD, AUD/USD, JPY/USD and RUB/USD exchange rates with a hybrid chaos-based SVR model. In their application, they confirm the forecasting superiority of their proposed model compared to chaos-based NNs and several traditional non-linear models. Lin and Pai (2010) introduce a fuzzy SVR model for forecasting indices of business cycles. Yuan (2012) suggests that (GA-εSVR) is more efficient than traditional SVR and NN models, when applied to the task of forecasting sales volume.

A more complicated but also promising class of SVRs is the Locally weighted SVR (LSVR). The conceptual advantage of the LSVR against the traditional SVR is the better balance between training error and model complexity. This is achieved by penalizing past data, while exploiting more the information from recent observations. This adaptive feature seems advantageous for financial time series. For example, Huang *et al.* (2006) apply LSVR to the task of forecasting three stock indices. Yang *et al.* (2009) apply a localized SVR forecast financial data and their results are superior to the standard SVR. Wu and Akbarov (2011) apply successfully weighted SVRs to the task of forecasting warranty claims. Finally, Jiang and He (2012) propose a hybrid SVR that incorporates the Grey relational grade weighting function. When applied to financial time series forecasting, the local Grey SVR outperforms locally weighted counterparts in terms of computational speed and prediction accuracy.

The above background motivates us to introduce the hybrid Locally Weighted Krill Herd - Support Vector Regression (KH-LSVR) algorithm. The KH algorithm tunes the three parameters of the LSVR. This hybrid structure, that combines the advantages of KH and LSVR, does not exist in the literature. In the few studies that employ LSVR the tuning of the LSVR is utilized through cross validation or grid search.

Previous studies in financial forecasting problems either use traditional optimized SVRs (Trafalis and Ince, 2000; Hsu *et al.*, 2009) or apply a simple GA in the SVR's parametrization (Pai *et al.*, 2006; Wu *et al.*, 2007; Yuan, 2012). In this study, the most popular SVR techniques applied in the relevant literature will act as benchmark to the proposed KH-LSVR algorithm. Namely, the performance of the KH-LSVR is compared against ε-SVR and ν-SVR algorithms optimized through a 5-fold cross-validation (εSVR_1 and νSVR_1) and grid-search techniques (εSVR_2 and νSVR_2). Additionally the proposed models will be benchmarked against genetically optimized ε-SVR and ν-SVR models.

All models are applied to the task of forecasting and trading seven Exchange-Traded Funds (ETFs) on a daily basis over the period 2006-2012. ETFs offer investors the opportunity to trade stock indices with low transaction costs. The seven ETFs under study track some of the most liquid US, European and emerging stock markets. In the GA and KH models the practitioner can choose the metaheuristics fitness function. In the literature, practitioners usually choose the Mean Square Error (MSE) or the Root Mean Square Error (RMSE). However, in financial trading applications statistical accuracy is not always synonymous of trading profitability. In this study, three different fitness functions will be explored incorporating statistical and trading terms. Additionally, in this research a novel SVR input selection process is introduced. The SVR input selection process is crucial for the performance of the model. In the introduced algorithm, at a first stage large pool of potential inputs (individual forecasts) is generated in the in-sample. Then, the Model Confidence Set (MCS) procedure proposed by Hansen *et al.* (2011) is applied in order to select the inputs with the highest informational capacity. In the relevant literature, SVR inputs are usually selected through cross validation or grid search. Both approaches are time consuming and lack the theoretical background of the MCS.

The forecasting and trading exercise covers the global financial crisis and the European sovereign debt crisis. Its aim is to test the performance of the models under study when the markets are in crises, to explore their robustness within the out-of-sample and to provide empirical evidence in favour of Adaptive Market Hypothesis (AMH), as presented by Lo (2004). For this reason, four forecasting and trading exercises over the period of 2006-2008 and 2010-2012 are conducted. In each forecasting exercise, the models under study are evaluated in a monthly basis. AMH suggests that the trading models' strength varies depending on market conditions and that the performance of trading rules dies out through time. It also suggests that in periods of financial crises it is more difficult to generate profitable trading rules and that this task is even more difficult when the underlying market is advanced. The monthly evaluation of our models, the out of sample periods and the markets under study will allow us to check these elements of the AMH.

In terms of the empirical results, the KH-LSVR outperforms its benchmarks in terms forecasting accuracy and trading profitability. The implementation of GAs in the SVR structures is beneficial in comparison with those applying data driven parametrization. The majority of the SVR models produce profitable forecasts after transaction costs, but their success seems sensitive to the parameter optimization and the periods under study. We note that the average trading performance of all models appears worse in the second forecasting exercise, while the profitability of all models deteriorates generally through time. The models generate lower profits through US ETFs in the first two forecasting exercises than the last two ones. The trend is opposite when it comes to ETFs tracking the EU market. Finally, all models perform better on the iShares MSCI Emerging Markets ETF.

The rest of the paper is organized as follows. Section 2 provides a detailed description of the dataset while a theoretical background on SVR and LSVR is given in section 3. The KH-LSVR algorithm and its benchmarks follow in section 4. The statistical and trading performance of all models is presented in sections 5 and 6 respectively while the concluding remarks are provided in section 8. Finally, the technical characteristics of the models under study are included in the appendix section.

2. Dataset

In this analysis we examine seven ETFs that are designed to replicate major stock indices from US, Europe and Emerging Markets over the periods of 2006-2008 and 2010-2012. ETFs, though, offer investors the opportunity to trade stock market indices at very low transaction costs[1]. The advantages of ETFs over "conventional trading" are well documented by researchers (Dolvin, 2010; Marshall *et al.*, 2013), practitioners (Ferri, 2009; Wagner, 2011), analysts (e.g. ETFdb.com) and institutions (e.g. ICI, US Securities and Exchange Commission (SEC)). The details of the seven ETFs under study are presented in table 1 below.

Table 1: The ETFs under study

MARKETS	ETF	TRACKING INDEX	TICKER
US	SPDR S&P 500 ETF Trust	S&P 500	SPY
	PowerShares QQQ Trust	NASDAQ-100	QQQ
	SPDR Dow Jones Industrial Average ETF Trust	Dow Jones Industrial Average	DIA
EU	SPDR EURO STOXX 50 ETF	EURO STOXX 50 Index	FEZ
	Vanguard FTSE Europe ETF	FTSE Developed Europe Index	VGK
	iShares MSCI Germany ETF	MSCI Germany Index	EWG
EMERGING	iShares MSCI Emerging Markets ETF	MSCI Emerging Markets Index	EEM

The seven ETF time series are non-normal and non-stationary, while they present negative skewness and positive kurtosis.[2]In order to overcome the non-stationarity issue, all series are transformed into daily series of rate returns using the following formula:

$$R_t = \ln\left(\frac{CP_t}{CP_{t-1}}\right) \tag{1}$$

where R_t is the rate of return and CP_t is the closing price (adjusted for dividends and stock splits) at time t. The descriptive statistics of the return series are shown in the following table:

[1] The transaction costs for the three ETFs tracking their respective benchmarks do not exceed 0.5% per annum for medium size investors (see, for instance, www.interactive-brokers.com). Before the expansion of ETFs, traders had to pay a separate commission for each individual stock of an industry-specific portfolio. Now there are sector-specific ETFs, which allow traders to pay only one commission to buy or sell short an entire group of stocks.

[2] The Jarque-Bera statistics (1980) confirm their non-normality at the 99% confidence interval.

Table 2: Summary statistics

	Ticker	SPY	QQQ	DIA	FEZ	VGK	EWG	EEM
2006-2008	Mean	-0.00036	-0.00041	-0.00019	-0.00018	-0.00029	-0.00001	-0.00020
	Standard deviation	0.0165	0.0169	0.0156	0.0205	0.0190	0.0203	0.0298
	Skewness	0.1090	0.0572	0.6686	0.4008	-0.1792	0.7025	0.3324
	Kurtosis	17.6295	10.7251	19.0925	15.6143	13.8570	18.3590	12.5684
	Jarque-Bera (p value)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	ADF (p value)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2010-2012	Mean	0.00039	0.00049	0.00038	-0.00012	0.00013	0.00020	0.00012
	Standard deviation	0.0116	0.0124	0.0105	0.0213	0.0185	0.0194	0.0166
	Skewness	-0.4384	-0.2873	-0.4173	-0.1563	-0.2733	-0.4383	-0.2790
	Kurtosis	6.5673	5.4327	6.3288	5.3138	5.9277	5.9534	5.8785
	Jarque-Bera (p value)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	ADF (p value)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

All returns series exhibit small skewness and high kurtosis. The Jarque-Bera statistic confirms that the seven return series are non-normal at the 99% confidence level. The Augmented Dickey-Fuller (ADF) reports that the null hypothesis of a unit root is rejected at the 99% statistical level for all ETFs.

The proposed methodology and its benchmarks are applied on the task of forecasting and trading the one day ahead rate of return ($E(R_t)$) of the seven ETFs. More specifically, the models' performance is evaluated through the four forecasting exercises presented in table 3.

Table 3: The total dataset

Note: The in-sample periods are the sum of the training and test datasets. The datasets of forecasting exercises F2 and F4 are simply formed by rolling the dataset of forecasting exercises F1 and F3 periods respectively by six months.

FORECASTING EXERCISE	PERIODS	TRADING DAYS	START DATE	END DATE
F1	Total Dataset	627	03/01/2006	30/06/2008
	Training Dataset	375	03/02/2006	29/06/2007
	Test Dataset	127	02/07/2007	31/12/2007
	Out-of-sample Dataset	125	02/01/2008	30/06/2008
F2	Total Dataset	630	03/07/2006	31/12/2008
	Training Dataset	377	03/07/2006	31/12/2007
	Test Dataset	125	02/01/2008	30/06/2008
	Out-of-sample Dataset	128	01/07/2008	31/12/2008
F3	Total Dataset	629	04/01/2010	29/06/2012
	Training Dataset	377	04/01/2010	30/06/2011
	Test Dataset	127	01/07/2011	30/12/2011
	Out-of-sample Dataset	125	03/01/2012	29/06/2012
F4	Total Dataset	630	01/07/2010	31/12/2012
	Training Dataset	380	01/07/2010	30/12/2011
	Test Dataset	125	03/01/2012	29/06/2012
	Out-of-sample Dataset	125	02/07/2012	31/12/2012

The four forecasting exercises cover the peaks of the global financial crisis (2008) and the European sovereign debt crisis (2012). The intuition behind the selection of the dataset is fourfold. Firstly, the performance of all models under stressed market conditions will be examined. Secondly, this performance will be compared between the different forecasting exercises. US markets in 2008 were in the centre of the turbulence, while during the European sovereign debt crisis they were not affected to the extent of the EU markets. On the other hand, the emerging markets were affected during both crises but not to the extent of the advanced markets. Thirdly, the robustness of the proposed models under different markets conditions will be examined. Lastly, detecting a connection between the empirical results with the AMH proposal will be attempted. Figure 1 presents the performance of the seven ETFs during the four forecasting exercises.

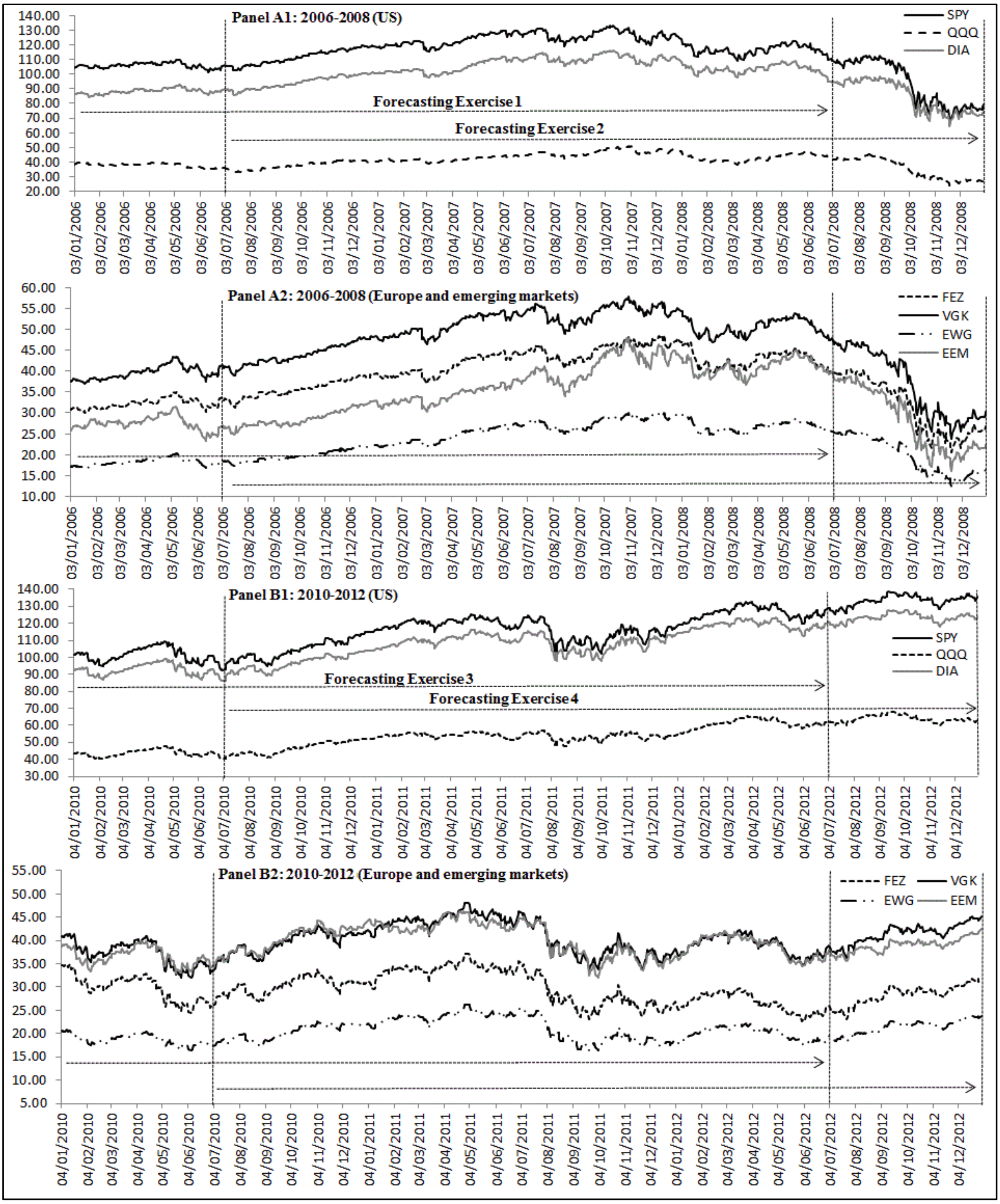


Figure 1: The ETFs under study

Note: Panels A1 and A2 show the first two forecasting exercises that investigate the period 2006-2008. Panels B1 and B2 present the other two forecasting exercises over the period of 2010-2012. Panels A1 and B1 refer to the US ETFs, while panels A2 and B2 include the European and Emerging Markets ETFs. All time series are ETFs' closing prices, adjusted for dividends and stock splits.

All models will be optimized in the in-sample and their forecasts will be evaluated in the out-of-sample.

3. Theoretical Framework

Support Vector Machines (SVMs) are learning machines utilizing the structural risk minimization principle to obtain good generalization on limited number of learning patterns (Wu and Liu, 2007). One class of SVM methods is the Support Vector Regression (SVR), introduced by Vapnik (1995), which is established as a robust technique for constructing data-driven and non-linear empirical regression models.

3.1 *ε*-SVR and v-SVR

Considering the training data $\{(x_1,y_1), (x_2,y_2)\ldots, (x_n, y_n)\}$, where $x_i \in X \subseteq R, y_i \in Y \subseteq R, i = 1...n$ and n the total number of training samples, then the SVR function can be specified as:

$$f(x) = w^T \varphi(x) + b \qquad (2)$$

$\varphi(x)$ is the non-linear function that maps the input data vector x into a feature space where the training data exhibit linearity (see figure 2c) while w and b are estimated by minimizing the regularized risk function:

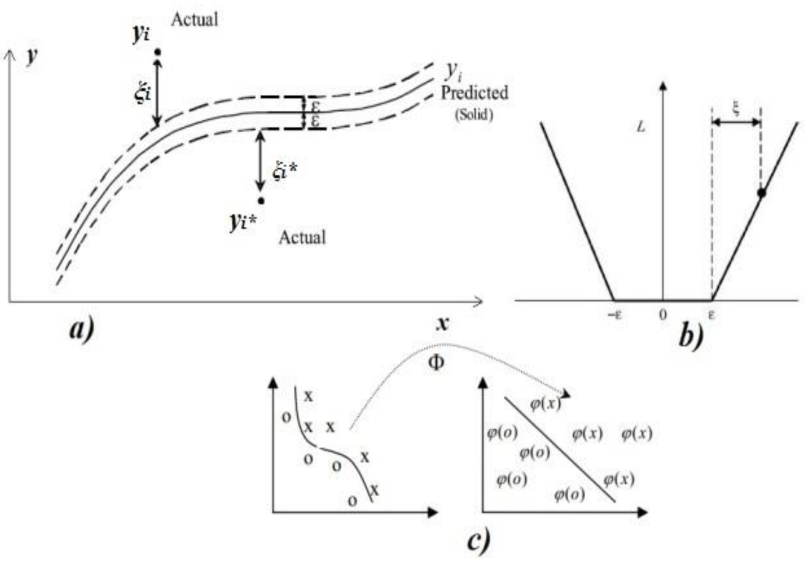
$$R(C) = C\frac{1}{n}\sum_{i=1}^{n} L_\varepsilon(y_i, f(x_i)) + \frac{1}{2}\|w\|^2 \qquad (3)$$

The parameters C and ε are predefined by the practitioner, y_i is the actual value at time i and $f(x_i)$is the predicted value at the same period. The ε-sensitive loss L_ε function (see figure 2b) is defined as:

$$L_\varepsilon(y_i, f(x_i)) = \begin{cases} 0 \;\; if \; |y_i - f(x_i)| \le \varepsilon \\ |y_i - f(x_i)| - \varepsilon \;\; if \;\; other \end{cases}, \varepsilon \ge 0 \qquad (4)$$

Equation (4) identifies the predicted values that have at most ε deviations from the actual values y_i. The ε parameter defines a ‘tube’ (see figure 2a). The two variables, ξ_i and ξ_i^* represent the distance of the actual values from the upper and lower bound of the ‘tube’ respectively.

Figure 2: a) The ε-tube b) The plot of the ε-sensitive loss function c) The mapping procedure by φ(x)



The goal is to solve the following argument:

$$\text{Minimize } C\sum_{i=1}^{n}(\xi_i+\xi_i^*)+\frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix}\xi_i \geq 0\\ \xi_i^* \geq 0\\ C>0\end{Bmatrix} \text{ and } \begin{Bmatrix} y_i - w^T\varphi(x_i)-b \leq +\varepsilon+\xi_i \\ w^T\varphi(x_i)+b-y_i \leq +\varepsilon+\xi_i^* \end{Bmatrix} \tag{5}$$

The quadratic optimization problem of equation (5) is transformed into a dual problem and its solution is based on the introduction of two Lagrange multipliers a_i, a_i^* and mapping with a kernel function $K(x_i, x)$:

$$f(x)=\sum_{i=1}^{n}(a_i-a_i^*)K(x_i,x)+b \text{ where } 0 \leq a_i, a_i^* \leq C \tag{6}$$

The application of the kernel function transforms the original input space into one with more dimensions, where a linear decision border can be identified. In this study, the Gaussian Radial Basis Function (RBF) for all SVRs is applied. A RBF kernel is in general specified as:

$$K(x_i,x)=\exp(-\gamma\|x_i-x\|^2), \gamma>0 \tag{7}$$

where γ is the variance of the kernel function.

RBFs require only one parameter to be optimized (the γ) and provide good forecasting results in similar SVR applications (see amongst others Lu *et al.*, (2009), Yeh *et al.*, (2011) and Kao *et al.* (2013)).[3]

Factor b in equation (6) is computed following the Karush-Kuhn-Tucker conditions. A detailed mathematical explanation of the solution can be found in Vapnik (1995). Support Vectors (SVs) are called the x_i's that lie closest to the ε margin, whereas non-SVs lie within the ε-tube. Increasing ε leads to more SVs' selection, whereas decreasing it results to more 'flat' estimates. The norm term $\|w\|^2$ characterizes the complexity (flatness) of the model and the term $\left\{\sum_{i=1}^{n}(\xi_i+\xi_i^*)\right\}$ is the training error, as specified by the slack variables. Consequently the introduction of parameter C satisfies the need to trade model complexity for training error and vice versa (Cherkassky and Ma, 2004).

The v-SVR algorithm can be used to make the optimization task easier, by encompassing the ε parameter in the optimization process and controls it with a new parameter $v \in (0,1)$. In v-SVR the optimization problem transforms to:

$$\text{Minimize } C\left(v\varepsilon+\frac{1}{n}\sum_{i=1}^{n}(\xi_i+\xi_i^*)\right)+\frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix}\xi_i \ge 0\\ \xi_i^* \ge 0\\ C>0\end{Bmatrix} \text{ and } \begin{Bmatrix} y_i - w^T\varphi(x_i)-b \le +\varepsilon+\xi_i \\ w^T\varphi(x_i)+b-y_i \le +\varepsilon+\xi_i^* \end{Bmatrix} \tag{8}$$

The methodology remains the same as in ε-SVR and the solution takes a similar form:

$$f(x)=\sum_{i=1}^{n}(a_i-a_i^*)K(x_i,x)+b \quad \text{where} \quad 0 \le a_i, a_i^* \le \frac{C}{n} \tag{9}$$

Based on the 'v-trick', presented by Schölkopf *et al.* (1999), increasing ε leads to the proportional increase of the first term of $\left\{v\varepsilon+\frac{1}{n}\sum_{i=1}^{n}(\xi_i+\xi_i^*)\right\}$, while its second term decreases proportionally to the fraction of points outside the ε-tube. So v can be considered as the upper bound on the fraction of errors. On the other hand, decreasing ε leads again to a proportional change of the first term, but also the second term's change is proportional to the fraction of SVs. That This means ε will shrink as long as the fraction of SVs is smaller than v, therefore v is also the lower bound in the fraction of SVs. For a more detailed mathematical analysis of solutions see Vapnik (1995).

[3] In this study, we have also experimented with the Wavelet kernel (Zhang *et al.*, 2004) and the Mahalanobis kernel (Ruiz and Lopez-de-Teruel, 2001). However, their use did not provide better results than the ones obtained with the application of the RBF kernel. For the shake of space, these results are not presented but are available upon request.

3.2. Locally Weighted Support Vector Regression

The locally weighted regression (LWR) is a memory-based procedure for fitting a regression surface to the data through multivariate smoothing. It is based on the assumption that the nearest to the predictor values are its best indicators. This is extremely beneficial in problems such as modelling financial trading series, where some training points are more important than others and more recent observations have higher weight in predicting the future.

LWR can approximate an estimate g(x) of the regression surface for every value x in the dimensional space of the independent variables. Following the suggestions of Cleveland and Devlin (1988) each point of the neighbourhood is weighted according to its distance from point of interest x. The neighborhood is set by estimating the distances of q observations x_i from x, where $1 \le q \le n$ Those points that are close to x are assigned large weights, while those that are far have small weights. This confirms the local element of the method (Lee et al., 2005).The idea of assigning weights to each point of the dataset could be expressed as:

$$\{(x_i, y_i, w_i)\}_{i=1}^{n}, x_i \in X \subseteq R, y_i \in Y \subseteq R,\ 0 \le w_i \le 1 \tag{10}$$

A quadratic function of the independent variables is fitted to the dependent variable using weighted least squares with these weights. In that way, g(x) is taken to be the value of this fitted function at x. A distance function in the space of the independent variables and a weight function to specify the neighborhood size are needed. The work of Cleveland and Devlin (1988) provides a detailed description on how to select these. The most common approach and the one followed in this study is to use the ratio *q/n* as a smoothness factor. The practitioner should interpret the smoothing factor rather than the *q*. The reason for that is that increasing the smoothing factor provides a smoother g(x) estimate. The selected weight function is the tricubic one specified below:

$$W(u) = \begin{Bmatrix} (1-u^3)^3, 0 \le u \le 1 \\ 0, otherwise \end{Bmatrix} \tag{11}$$

Based on these, the weight of each training data (x_i, y_i) is:

$$w_i(x_i) = W\left(\frac{\rho(x, x_i)}{d(x)}\right) \tag{12}$$

where ρ is the Euclidian distance and *d(x)* is the Euclidian distance specifically from the q^{th}-nearest x_i to x. [4]

[4] For instance, when the daily return for ETFs is desired and the expected return for the respective time series is roughly equal to zero, outliers of 5% gain per ticker could be ignored due to major structural change rather than routine behaviour of the time series. In this

From equation (11) it is verified that $w_i \in [0,1]$. The weight has its maximum value when x_i is closest to x and its minimum for the q^{th}-nearest x_i to x.

Applying the principles of LWR to the SVR, we can achieve a Locally Weighted SVR (LSVR), where the parameter C is not constant, but locally adjusted as: $C_i = w_i * C$ (13)

In the case of ε-SVR, the quadratic optimization problem of equation (5) is transformed to:

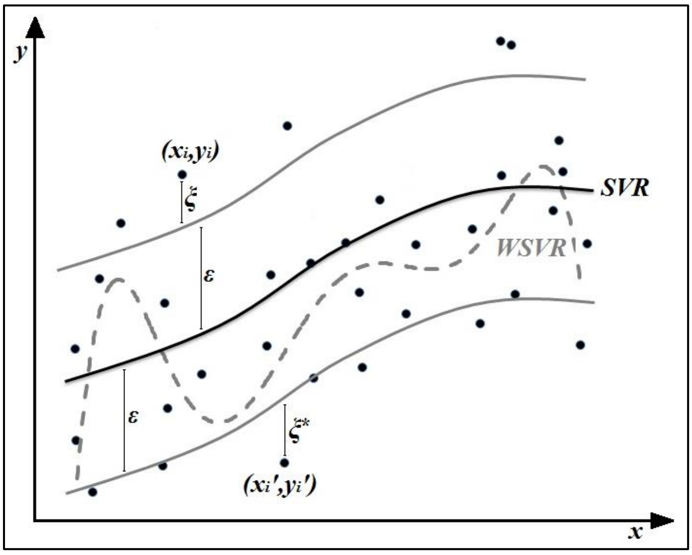
$$\text{Minimize } \sum_{i=1}^{n} C_i(\xi_i + \xi_i^*) + \frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix} \xi_i \geq 0 \\ \xi_i^* \geq 0 \\ C_i > 0 \end{Bmatrix} \text{ and } \begin{Bmatrix} y_i - w^T\varphi(x_i) - b \leq +\varepsilon + \xi_i \\ w^T\varphi(x_i) + b - y_i \leq +\varepsilon + \xi_i^* \end{Bmatrix} \quad (14)$$

Regarding v-SVR, the quadratic optimization problem of equation (8) becomes the following:

$$\text{Minimize } C_i\left(v\varepsilon + \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*)\right) + \frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix} \xi_i \geq 0 \\ \xi_i^* \geq 0 \\ C_i > 0 \end{Bmatrix} \text{ and } \begin{Bmatrix} y_i - w^T\varphi(x_i) - b \leq +\varepsilon + \xi_i \\ w^T\varphi(x_i) + b - y_i \leq +\varepsilon + \xi_i^* \end{Bmatrix} \quad (15)$$

The conceptual advantage of the LSVR against the traditional SVR is based on the locally adjusted C_i. The weighted C_i provides better balance between training error and model complexity, by penalizing more (small weight) big slack variables (see figure 3).

Figure 3: the ε-tube (grey lines), the f(x) curves of SVR (black solid line) and WSVR (grey dashed line)



The traditional SVR attempts to track all training data with a specific model complexity through a constant C. Conceptually, this suggests that that the size of ξ_i, ξ_i^* does not fluctuate much. The LSVR, on the other hand, highly penalizes the errors near x in an attempt to increase predictability. In that way, LSVR's predictive performance is increasing gradually, as the shape of the weight function is

example x is the expected return target when modelling the deviation from this point, the outlier is x_i and w_i is the weight the Euclidean distance assigns to this observation.

becoming sharper (Lee *et al.*, 2005).[5] This is a clear point of superiority of LSVR over the non-locally optimized method. Nonetheless, it should be clarified that the weighting process does not solve 'over-fitting' issues that usually impede the success of SVR applications.

4. Methodology

This section describes the proposed algorithm. Initially, the proposed hybrid model is introduced, while a summary description of its benchmarks follows. Finally, the input selection process for all models is presented.

4.1 Locally Weighted Krill Herd Support Vector Regression (KH-LSVR)

As discussed earlier, LSVR provides better balance between the training error and model complexity, which is crucial for the success of the SVR method. However, this does not provide a safety net to affront the major challenge of 'over-fitting'. The most common SVR optimization approaches, the cross-validation and grid search, are data and task biased (Zhang *et al.*, 1999).

For that reason, we propose a hybrid Krill Herd – Support Vector Regression that embodies a KH algorithm for optimal parameter selection to the LSVR process, as shown in section 3.2. The KH algorithm, as presented by Gandomi and Alavi (2012), is an innovative metaheuristic optimization technique that simulates the herding behavior of krill individuals. The intuition of the analysis is the mean-reversion effect of predators' attacks on the herd of krills. Such attacks result in the reduction of the krill density of the herd. After the attack, the herd must increase its density by sensing nearby krill but without deviating much from the optimal path to reach food.

Based on the above, Gandomi and Alavi (2012) propose that the position (P) of each krill in the search space is influenced by the movement induced by other krill (M), the foraging action (F) and the random diffusion (RD). All these motions can be summarized in one Lagrangian formulation for every krill j:

$$\frac{dP_j}{dt} = M_j + F_j + RD_j \tag{16}$$

The new movement motion M^{t+1} of each krill j is calculated as:

$$\begin{Bmatrix} M_j^{t+1} = M^{\max} eff_j + k_M \mathrm{M}^t \\ eff_j = eff_j^{loc} + eff_j^{t\arg} \end{Bmatrix} \tag{17}$$

[5] The weight of every point for a traditional regression model is 1/n, meaning that the assigned weight is similar for every point. In LWR the importance/weight increases continuously, once we move from outliers to more central points. Plotting the weight functions for both cases, it is obvious that in the traditional regression the function is a horizontal line, whereas in LWR the function is has a bell-shape around the central point. Once the importance of distance is increased through higher orders of power, the LWR function plot becomes narrower or in other words sharper.

Where:

- M^{max}: the maximum induced speed
- $k_M \in [0,1]$: the inertia weight of the motion
- M_j^t: the previous movement motion
- eff_j: the direction of the motion
- $eff_j^{loc}, eff_j^{t\arg}$:the local effect by neighbor krill and the target direction effect by the best individual krill.

Gandomi and Alavi (2012) suggest that the local search of the algorithm is based on an attractive/repulsive tendency between individual krills. The neighbor krills are identified through a sensing distance from the j^{th} one:

$$d_{s,j} = (1/\mathrm{N_k})\sum_{j'=1}^{N_k} \left\| P_j - P_{j'} \right\| \tag{18}$$

where N_k is the number of krill individuals.

The new foraging motion F^{t+1} of every krill j is also calculated on the basis of two factors, namely the food location and its previous experience in locating a correct food position:

$$\left\{ \begin{matrix} F_j^{t+1} = V_F\, floc_j + k_F F_j^t \\ floc_j = floc_j^{food} + floc_j^{best} \end{matrix} \right\} \tag{19}$$

where:

- V_F : the foraging speed[6]
- $k_F \in [0,1]$: the inertia weight of the motion
- F_j^t: the previous foraging motion
- $floc_j$: the location of the food
- $floc_j^{food}$, $floc_j^{best}$: the food attractive and the effect from the best food-locating j^{th} krill so far

The food attraction is defined to provide global optima for the krill swarm. The third motion RD of krills is calculated as a maximum diffusion speed RD^{max} and a random directional vector δ with values between -1 and 1. In other words: $RD = RD^{\max}\delta$ (20)

The equations (17) and (19) suggest the future krill motion towards the optimal position by performing two parallel local and global strategies, something that makes the KH algorithm very robust. Krill continue their local search (equation (17)) until the herd density increases. When that happens, more and more krill orientate to food (equation (19)) rather than the nearby krill. These two strategies provide the fitness values for several effective factors that induce an attractive/repulsive motion

[6] The maximum induced speed of equation (17) and the foraging speed of equation (19) are set to 0.01 and 0.02 ms^{-1} respectively, as Gandomi and Alavi (2012) suggest.

response to each krill. The equation (20) performs a random search in the proposed search space, diffusing any potential biased motion responses to the herd (either towards food locations or neighboring sensed krill). For more details on the approximation of these values, refer to the extensive mathematical steps of Gandomi and Alavi (2012).

The position *Pj* of each krill at time $t+\Delta t$ is given as:

$$\left\{ \begin{aligned} & P_j(t+\Delta t) = P_j(t) + \Delta t \frac{dP_j}{dt} \\ & \Delta t = Z_{cr} \sum_{r=1}^{NP} \left(UpB_r - LowB_r \right) \end{aligned} \right\} \tag{21}$$

where:

- $Z_{cr} \in [0,2]$: constant number
- NP: the number of parameters optimized (in our case NP=3)
- $UpB_r, LowB_r$: the upper and lower bounds of the parameters

The Δt practically is the only parameter that needs fine tuning. This is the striking advantage of the method compared to other more complicated metaheuristics approaches.

In the KH-LSVR optimization, the practitioner needs to predefine three parameters. The range of the bounds of each parameter defines the potential three-dimensional search space. The Z_{cr} is set at values lower than 1, because it allows careful search of the space by the krill individuals. Krill behavior suggests that herd individuals at an initial point (predator attack) tend to focus on exploration of the search space and then its exploitation. For that reason, k_M and k_F are initially set high (0.9). These parameters are linearly decreased to 0.1 at the end to encourage exploitation (Gandomi and Alavi, 2012). Next two genetic reproduction mechanisms (mutation and crossover) are implemented in order to further improve the performance of the krill positions.

The KH algorithm is optimized based on a fitness function. In similar applications of metaheuristics, the practitioners set as fitness function a simple statistical measure such as the MSE or the RMSE. However, in financial forecasting applications the trading performance is of the outmost importance. In this research, the algorithm is trained through three different fitness functions. This exercise should improve the forecasting performance of our models. The choice of the fitness function is based on the performance of the models in the in-sample. The first function aims to minimize the RMSE (equation (22)). The second one aims to maximize the annualized return (equation (23)) while the third (equation (24)) attempts to provide a balance between statistical accuracy and trading efficiency.

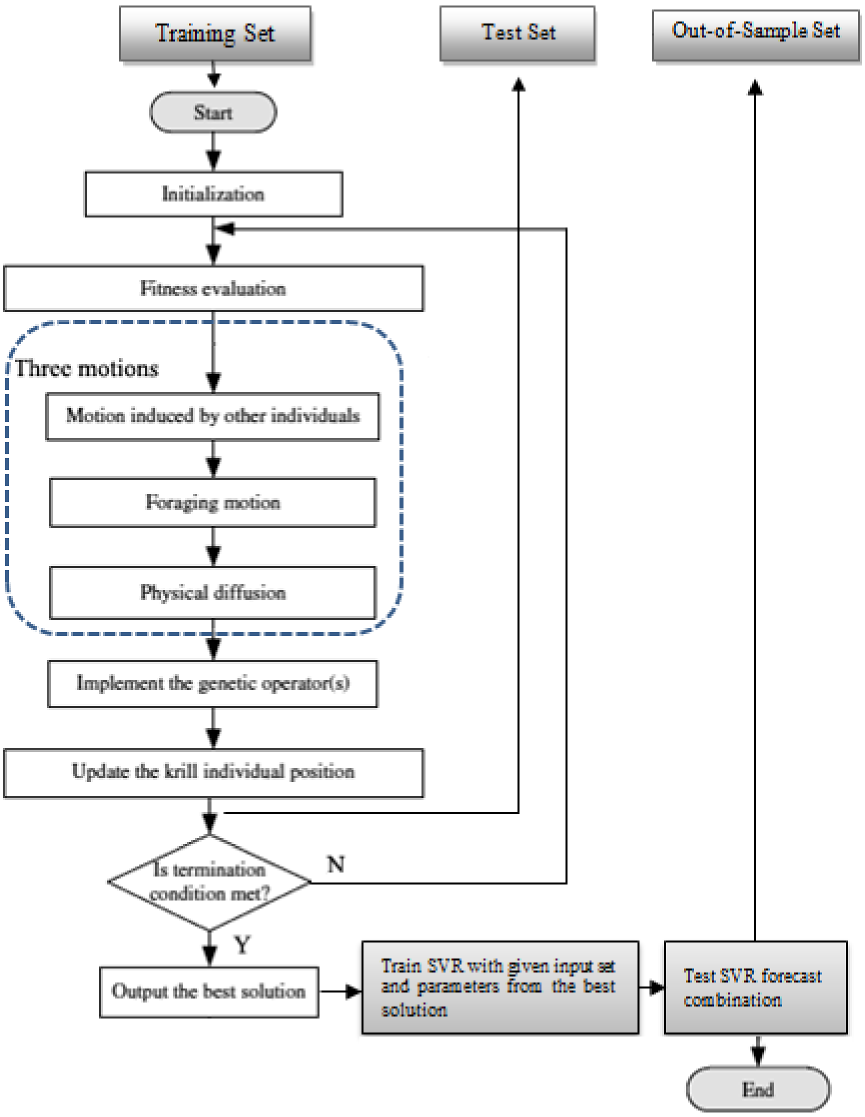
$$Fitness_1 = 1/(1+RMSE) \tag{22}$$

$$Fitness_2 = Annualized\ Return \tag{23}$$

$$Fitness_3 = Annualized\ Return - 10 * RMSE \quad (24)$$

The aim of the algorithm is to maximize equations (22), (23) and (24). The two KH algorithms are trained in the training sub-period and their performance is evaluated in the test sub-period. The fitness is evaluated in terms of Annualized Return. The function that performs better will be applied in the out-of-sample. The outcome of this process is presented in appendix B. We note that $Fitness_3$ seems to dominate the selection process. Nevertheless there are several cases that the other two functions are selected. These results outline the importance of experimenting on the metaheuristics fitness function in the in-sample. Following a priori selections can deter the performance of the metaheuristics algorithm. The flowchart of the proposed methodology is presented in figure 4 below, while the training characteristics of KH are summarized in appendix B.

Figure 4: KH-LSVR flowchart



4.2 Benchmark Models

The statistical and trading efficiency of the hybrid model is initially evaluated by benchmarking it with traditional non-genetically and genetically optimized SVRs. As it was discussed before, SVR practitioners face a constant tackle, the optimal selection of the model's parameters (C, ε or v and kernel function parameter). 5-fold cross validation and grid search are the most common statistical approaches while GA[7] dominates the metaheuristics literature on SVR optimization. Based on this fact as benchmarks of the KH Locally Weighted SVRs are selected:

- An ε-SVR model (εSVR_1) and an v-SVR model ($vSVR_1$) where all parameters are selected through 5-fold cross-validation.
- An ε-SVR model (εSVR_1) and v-SVR model ($vSVR_2$) where all parameters are selected through grid search.
- An ε-SVR model (GA-εSVR) and v-SVR model (GA-vSVR) where all parameters are selected through genetic algorithm.

4.3 Input Selection

In order to select the SVR inputs, a set of potential linear and non-linear predictors for the in-sample is generated. The linear pool includes Simple Moving Averages (SMA), Exponential Moving Averages (EMA), Autoregressive terms (AR), Autoregressive Moving Average (ARMA) models, Rate of Change Indicators (ROC), and a Pivot Point Indicator (PPI). The non-linear Smooth Transition Autoregressive Models (STAR), Nearest Neighbors Algorithms (k-NN), a Multi-Layer Perceptron (MLP), a Recurrent Neural Network (RNN), a Higher Order Neural Network (HONN), a Psi-Sigma Neural Network (PSN), Adaptive RBF and PSO Neural Network (ARBF-PSO), Genetic Programming (GP) and Gene Expression Programming (GEP) complete the input pool. These predictors create a pool of three hundred and forty eight individual predictors in total for each forecasting exercise and each ETF series. All predictors have been successfully applied in financial forecasting applications. The proposed models will combine the best predictors in order to generate superior out-of-sample performance. Combining individual linear and non-linear forecasts to improve forecast accuracy is common practice among practitioners and researchers (Timmermann, 2006; Park and Irwin, 2007). Forecast combinations generate more robust signals and in trading applications offer the benefits of model diversification. A short summary of the models that consist the input pool is provided in Appendix A.

The dimensions of the potential input vectors are large. In order to screen the optimal inputs and cope with the dimensionality issue, we perform the MCS procedure proposed by Hansen *et al.* (2011). The MCS test deduces the superior predictors from a full set of models, given specific criteria and confidence levels. The procedure provides a random data-dependent set of best forecasting models, acknowledging the information limitations in the given datasets. Hansen *et al.* (2011) suggest that the

[7] For more details on the GA algorithm see Holland (1975). Its characteristics are presented in Appendix B. Similarly as the KH, the GA algorithm is centered around a fitness function, The fitness function selection process that is presented on section 4.1 is also applied to the GA-εSVR and the GA-vSVR models. These results are also presented in Appendix B.

more informative the data are, the fewer models are included in the MCS. In this study, the MCS criterion is the MSE and the confidence level is set to 90%. Higher confidence level would limit the input set to only 1 or 2 models while a lower level would have included inputs not informationally efficient. The result of this process is presented in table 5.

Table 5: The SVR sets of inputs

	SPY	QQQ	DIA	FEZ	VGK	EWG	EEM
F1	MLP, HONN, PSN, GP	RNN, PSN, *k*-NN, GP, GEP	ARMA (2,7) , HONN, PSN, *k*-NN	ARMA (1,8), MLP, PSN, GP	EMA(5), RNN, *k*-NN, GP	MLP, ARBF-PSO, PSN, GP	RNN, PSN, *k*-NN, GEP
F2	HONN, RNN, ARBF-PSO, GP	RNN, PSN, HONN, GEP	MLP, PSN ARBF-PSO, GP	RNN, ARBF-PSO, *k*-NN, GEP	MLP, HONN, RNN, ARBF-PSO	HONN, RNN, PSN, *k*-NN	HONN, RNN, PSN, *k*-NN
F3	MLP, HONN, RNN, PSN	HONN, PSN, GP, GEP	SMA(8), MLP, HONN, GEP	ESTAR(8), MLP, ARBF-PSO, GEP	RNN, PSN, *k*-NN, GEP	HONN, ARBF-PSO, PSN, GEP	MLP, HONN, ARBF-PSO, GP
F4	MLP, RNN, PSN, GP	RNN, PSN, ARBF-PSO, *k*-NN	MLP, PSN, *k*-NN, GEP	MLP, RNN, PSN, *k*-NN,GP	MLP, HONN, RNN,PSN, GP	LSTAR(5), MLP, *k*-NN, GP, GEP	RNN, *k*-NN, GP,GEP

The above table shows that the previous two-step selection process vastly decreases the dimension of the final input vectors. On average, the number of selected inputs is 4 out of 348. From table 5, it is obvious that the non-linear models dominate the selected input sets. This is expected to some extent due to the non-linear nature of the time series under study.

5. Statistical Evaluation

This section provides the out-of-sample statistical performance of all models applied. In 5.1 the statistical accuracy of the proposed models is presented, while in 5.2 the genuineness of the forecasts is evaluated through the Giacomini and White (GW) (2006) test.

5.1 Statistical Accuracy

The statistical accuracy of the obtained forecasts is evaluated through the RMSE (see appendix C), the Pesaran-Timmermann (PT) (1992) test and the Diebold-Mariano (DB) (1995) test. The PT test is used to examine whether the directional movements of the real and forecast values are in step with one another. The null hypothesis is that the model under study has no power on forecasting the relevant ETF return series. The DB test checks the null hypothesis of equal predictive accuracy between our models' forecasts[8]. For more details on the PT and the DB test see Pesaran and Timmermann (1992) and Diebold and Mariano (1995) respectively. The out-of-sample results are summarized in table 6.

Table 6: Out-of-sample Statistical Performance

*Note: The table reports the RMSE values of each SVR forecast while the PT statistics are in the parenthesis. ***, ** and * denotes that the DM null hypothesis is rejected at the 1%, 5% and 10% significance level respectively. In bold are the lowest RMSE values in each forecasting exercise.*

	ETF	εSVR$_1$	εSVR$_2$	vSVR$_1$	vSVR$_2$	GA-εSVR	GA-vSVR	KH-LSVR$_1$	KH-LSVR$_2$
F1	SPY	0.0061 (5.15)***	0.0059 (5.35)***	0.0059 (6.25)***	0.0060 (7.34)**	0.0057 (7.57)***	0.0053 (7.88)***	0.0052 (8.39)*	**0.0048** (8.58)

[8] In our exercise we apply the DB test to couples of forecasts (KH-LSVR$_2$ vs. another forecasting model). A rejection of the null hypothesis suggests that the first forecast (the KH-LSVR$_2$) is more accurate.

	QQQ	0.0063 (4.67)***	0.0062 (5.52)**	0.0058 (6.05)***	0.0059 (6.42)***	0.0056 (7.23)**	0.0056 (7.84)***	0.0053 (8.50)*	**0.0051** (9.25)
	DIA	0.0060 (5.33)***	0.0060 (6.28)***	0.0058 (7.14)***	0.0057 (7.82)**	0.0057 (8.37)**	0.0056 (8.56)**	0.0052 (9.32)*	**0.0049** (10.07)
	FEZ	0.0065 (5.45)***	0.0063 (6.04)***	0.0063 (6.72)**	0.0061 (6.15)***	0.0059 (7.43)***	0.0059 (8.04)***	0.0057 (8.72)*	0.0053 (9.36)
	VGK	0.0061 (4.31)***	0.0059 (4.83)***	0.0057 (5.67)***	0.0054 (6.46)***	0.0053 (7.12)**	0.0050 (7.55)**	**0.0046** (7.86)	0.0047 (8.87)
	EWG	0.0068 (4.65)***	0.0065 (5.28)***	0.0064 (5.87)**	0.0062 (6.65)**	0.0061 (7.25)**	0.0061 (7.94)***	**0.0058** (8.48)	0.0059 (8.74)
	EEM	0.0065 (5.44)***	0.0063 (6.62)***	0.0061 (7.35)***	0.0059 (7.86)***	0.0057 (8.42)***	0.0054 (8.83)***	0.0053 (9.29)*	**0.0050** (9.92)
F2	SPY	0.0063 (4.22)***	0.0060 (4.48)***	0.0061 (5.42)***	0.0061 (6.55)***	0.0059 (7.05)***	0.0056 (7.39)**	0.0054 (8.27)*	**0.0053** (8.50)
	QQQ	0.0065 (4.31)***	0.0065 (5.18)***	0.0062 (5.76)***	0.0061 (5.97)**	0.0059 (6.28)**	0.0059 (7.78)**	0.0056 (8.19)	**0.0054** (8.51)
	DIA	0.0064 (4.79)***	0.0065 (5.44)***	0.0063 (5.99)**	0.0061 (6.57)***	0.0059 (7.32)**	0.0059 (7.47)***	0.0056 (8.57)	**0.0055** (9.42)
	FEZ	0.0067 (4.38)***	0.0067 (5.37)***	0.0066 (5.79)***	0.0063 (6.73)**	0.0061 (7.41)***	0.0060 (7.56)**	0.0058 (8.35)*	0.0055 (8.67)
	VGK	0.0065 (4.05)**	0.0065 (4.25)***	0.0062 (5.29)**	0.0064 (6.12)***	0.0061 (6.57)**	0.0058 (7.37)***	0.0057 (7.68)*	**0.0054** (8.79)
	EWG	0.0075 (4.53)***	0.0072 (5.06)***	0.0069 (5.58)***	0.0070 (6.06)***	0.0068 (6.75)***	0.0067 (7.57)***	0.0064 (8.67)*	**0.0062** (8.89)
	EEM	0.0071 (5.14)***	0.0073 (5.65)**	0.0071 (6.53)***	0.0070 (7.07)**	0.0067 (7.33)***	0.0065 (8.41)***	0.0061 (9.08)*	**0.0057** (9.35)
F3	SPY	0.0059 (5.42)***	0.0057 (5.77)***	0.0055 (6.65)***	0.0057 (7.79)***	0.0055 (8.34)***	0.0051 (8.97)**	0.0051 (9.57)*	**0.0044** (10.24)
	QQQ	0.0061 (5.05)***	0.0060 (5.74)***	0.0060 (6.27)***	0.0057 (7.32)**	0.0058 (7.75)**	0.0054 (8.39)***	0.0051 (9.05)*	**0.0048** (9.89)
	DIA	0.0054 (5.55)***	0.0052 (6.41)**	0.0052 (7.04)***	0.0051 (7.33)***	0.0049 (8.77)**	0.0047 (9.14)***	**0.0039** (9.54)	0.0040 (10.12)
	FEZ	0.0061 (5.87)***	0.0062 (6.34)***	0.0059 (6.81)***	0.0058 (7.36)***	0.0056 (7.97)***	0.0055 (8.75)**	0.0055 (9.27)*	**0.0049** (10.05)
	VGK	0.0055 (4.75)***	0.0053 (5.37)***	0.0052 (5.98)**	0.0052 (6.81)***	0.0050 (7.39)**	0.0047 (7.74)**	0.0048 (9.04)*	**0.0043** (10.33)
	EWG	0.0061 (5.42)***	0.0060 (6.07)***	0.0057 (6.93)***	0.0056 (7.71)***	0.0055 (8.49)***	0.0054 (9.07)***	0.0051 (9.44)*	**0.0047** (10.47)
	EEM	0.0062 (5.94)**	0.0062 (6.25)***	0.0058 (7.01)***	0.0059 (7.42)**	0.0056 (8.53)***	0.0055 (9.25)**	0.0051 (9.77)*	**0.0046** (10.24)
F4	SPY	0.0057 (5.19)***	0.0057 (5.38)***	0.0056 (5.67)**	0.0052 (6.62)***	0.0051 (7.61)**	0.0049 (8.45)***	**0.0048** (9.02)*	0.0047 (9.73)
	QQQ	0.0063 (4.58)***	0.0064 (5.47)***	0.0060 (6.31)***	0.0059 (6.86)***	0.0058 (7.53)***	0.0057 (8.13)**	0.0055 (8.83)*	**0.0052** (9.45)
	DIA	0.0054 (4.98)***	0.0051 (5.44)**	0.0050 (5.73)**	0.0050 (6.87)***	0.0047 (7.54)***	0.0046 (8.75)***	0.0041 (8.92)*	**0.0039** (9.85)
	FEZ	0.0066 (4.42)***	0.0065 (5.57)***	0.0063 (5.79)***	0.0064 (6.73)***	0.0060 (6.99)**	0.0058 (8.57)***	0.0057 (9.05)	**0.0051** (10.08)
	VGK	0.0062 (3.84)***	0.0058 (4.77)***	0.0055 (5.43)***	0.0056 (6.02)***	0.0052 (6.93)***	0.0050 (7.96)***	0.0049 (8.69)*	**0.0046** (9.96)
	EWG	0.0060 (4.67)***	0.0058 (5.41)**	0.0057 (6.48)***	0.0054 (7.04)***	0.0052 (7.63)**	0.0052 (8.42)***	0.0048 (9.44)*	**0.0041** (10.07)
	EEM	0.0063 (4.91)***	0.0060 (5.69)***	0.0057 (5.99)***	0.0057 (7.03)***	0.0054 (7.87)***	0.0053 (8.41)***	0.0051 (8.85)*	**0.0044** (9.39)

The above results show that the traditional SVRs provide less accurate forecasts in comparison to their GA and KH counterparts. The proposed KH-LSVRs outperform those employing GAs in the vast majority of cases. Overall, the 'v-trick' leads to a boost of forecasting accuracy between models of same SVR class. In terms of RMSE, the KH-LSVR$_2$ provides the lowest values in most of the cases. The DM test also demonstrates the superiority of the KH-LSVR$_2$ forecasts in almost all ETFs and periods under study. It should be noted though, that the DM statistics between the two KH LSVRs are not always significant. This indicates that the KH-LSVR results are more likely to be driven by the benefits of the KH parameter optimization applied in the local SVR process, rather than the use of the SVR 'v-trick' per se. The significant PT statistics reveal that all SVR models are capable of capturing the directional movements of the seven ETF return series. The statistical accuracy of the forecasts deteriorates during F1 and F2 periods. In particular, the worst out-of-sample results are obtained in F2, where the peak of the global financial crisis is observed.

5.2. Giacomini-White test

The previous statistical results are further authenticated by computing the unconditional GW test for the out-of-sample predictive ability testing and forecast selection. The null hypothesis of the GW test is the equivalence in forecasting accuracy between two forecasting models. The sign of the test statistic specifies the superior model according to its forecasting performance. A positive realization of the GW test statistic indicates that the second model is more accurate than the first one whereas a negative specifies the opposite. The test is calculated based on the MSE loss function. The outcomes of the GW test are presented in table 7 below.

Table 7: The Giacomini-White test for all out-of-sample periods.

*Note: The table displays the p-values of the statistic under the null hypothesis that the column model shows equivalent performance compared with each row model for every ETF separately. ***and ** denote a rejection of the null hypothesis at the 1% and 5%significance level respectively.*

	ETF	Models	εSVR$_1$	εSVR$_2$	vSVR$_1$	vSVR$_2$	GA-εSVR	GA-vSVR
F1	SPY	KH-LSVR$_1$	0.000***	0.000***	0.005***	0.016**	0.025**	0.029**
		KH-LSVR$_2$	0.000***	0.000***	0.002***	0.006***	0.008***	0.011**
	QQQ	KH-LSVR$_1$	0.001***	0.001***	0.006***	0.008***	0.015**	0.017**
		KH-LSVR$_2$	0.000***	0.000***	0.001***	0.005***	0.007***	0.012**
	DIA	KH-LSVR$_1$	0.001***	0.005***	0.013**	0.022**	0.041**	0.047**
		KH-LSVR$_2$	0.000***	0.002***	0.004***	0.009***	0.019**	0.032**
	FEZ	KH-LSVR$_1$	0.000***	0.000***	0.002***	0.002***	0.007***	0.008***
		KH-LSVR$_2$	0.000***	0.000***	0.000***	0.003***	0.008***	0.007***
	VGK	KH-LSVR$_1$	0.000***	0.003***	0.003***	0.018**	0.037**	0.041**
		KH-LSVR$_2$	0.000***	0.001***	0.000***	0.004***	0.016**	0.029**
	EWG	KH-LSVR$_1$	0.001***	0.000***	0.005***	0.016**	0.028**	0.033**
		KH-LSVR$_2$	0.000***	0.000***	0.000***	0.006***	0.023**	0.028**
	EEM	KH-LSVR$_1$	0.000***	0.000***	0.002***	0.003***	0.006***	0.006***
		KH-LSVR$_2$	0.000***	0.001***	0.000***	0.001***	0.004***	0.004***
F2	SPY	KH-LSVR$_1$	0.012**	0.005***	0.011**	0.014**	0.032**	0.036**
		KH-LSVR$_2$	0.000***	0.000***	0.004***	0.016**	0.021**	0.025**
	QQQ	KH-LSVR$_1$	0.003***	0.008***	0.012**	0.018**	0.023**	0.040**
		KH-LSVR$_2$	0.000***	0.000***	0.004***	0.009***	0.016**	0.023**
	DIA	KH-LSVR$_1$	0.002***	0.002***	0.006***	0.020**	0.024**	0.037**
		KH-LSVR$_2$	0.000***	0.000***	0.001***	0.013**	0.015**	0.029**
	FEZ	KH-LSVR$_1$	0.007***	0.012**	0.012**	0.021**	0.026**	0.043**
		KH-LSVR$_2$	0.004***	0.008***	0.014**	0.016**	0.019**	0.030**
	VGK	KH-LSVR$_1$	0.000***	0.005***	0.004***	0.006***	0.008***	0.017**
		KH-LSVR$_2$	0.000***	0.000***	0.002***	0.003***	0.008***	0.014**
	EWG	KH-LSVR$_1$	0.001***	0.005***	0.013**	0.012**	0.038**	0.035**
		KH-LSVR$_2$	0.000***	0.000***	0.005***	0.005***	0.006***	0.009***
	EEM	KH-LSVR$_1$	0.005***	0.014**	0.012**	0.019**	0.040**	0.042**
		KH-LSVR$_2$	0.007***	0.008***	0.008***	0.021**	0.031**	0.036**
	SPY	KH-LSVR$_1$	0.002***	0.003***	0.008***	0.015**	0.019**	0.041**
		KH-LSVR$_2$	0.000***	0.000***	0.002***	0.003***	0.014**	0.016**
	QQQ	KH-LSVR$_1$	0.000***	0.002***	0.006***	0.008***	0.007***	0.041**
		KH-LSVR$_2$	0.000***	0.001***	0.002***	0.005***	0.006***	0.025**
	DIA	KH-LSVR$_1$	0.001***	0.008***	0.013**	0.015**	0.022**	0.036**
		KH-LSVR$_2$	0.000***	0.004***	0.006***	0.011**	0.014**	0.027**

F3	**FEZ**	**KH-LSVR$_1$**	0.002***	0.000***	0.005***	0.006***	0.005***	0.009***
		KH-LSVR$_2$	0.000***	0.000***	0.002***	0.008***	0.008***	0.012**
	VGK	**KH-LSVR$_1$**	0.014**	0.012**	0.017**	0.023**	0.027**	0.036**
		KH-LSVR$_2$	0.012**	0.016**	0.022**	0.018**	0.021**	0.030**
	EWG	**KH-LSVR$_1$**	0.002***	0.000***	0.004***	0.006***	0.005***	0.006***
		KH-LSVR$_2$	0.000***	0.000***	0.002***	0.005***	0.005***	0.006***
	EEM	**KH-LSVR$_1$**	0.000***	0.000***	0.007***	0.018**	0.009***	0.033**
		KH-LSVR$_2$	0.000***	0.001***	0.000***	0.002***	0.005***	0.011**
F4	**SPY**	**KH-LSVR$_1$**	0.000***	0.000***	0.001***	0.005***	0.004***	0.007***
		KH-LSVR$_2$	0.000***	0.000***	0.000***	0.000***	0.002***	0.005***
	QQQ	**KH-LSVR$_1$**	0.000***	0.003***	0.008***	0.020**	0.016**	0.025**
		KH-LSVR$_2$	0.000***	0.006***	0.009***	0.015**	0.017**	0.016**
	DIA	**KH-LSVR$_1$**	0.002***	0.012**	0.014**	0.006***	0.008***	0.020**
		KH-LSVR$_2$	0.000***	0.000***	0.006***	0.008***	0.011**	0.016**
	FEZ	**KH-LSVR$_1$**	0.012**	0.008***	0.015**	0.011**	0.008***	0.013**
		KH-LSVR$_2$	0.007***	0.005***	0.008***	0.014**	0.012**	0.019**
	VGK	**KH-LSVR$_1$**	0.000***	0.000***	0.004***	0.008***	0.024**	0.042**
		KH-LSVR$_2$	0.000***	0.000***	0.0013**	0.006***	0.019**	0.012**
	EWG	**KH-LSVR$_1$**	0.001***	0.001***	0.005***	0.006***	0.009***	0.015**
		KH-LSVR$_2$	0.000***	0.001***	0.001***	0.003***	0.005***	0.006***
	EEM	**KH-LSVR$_1$**	0.000***	0.002***	0.017**	0.026**	0.041**	0.038**
		KH-LSVR$_2$	0.000***	0.000***	0.001***	0.003***	0.005***	0.020**

The above results further validate the statistical findings of the previous section. KH-LSVRs are found to be statistically superior to their traditional and genetically optimized benchmarks, since the null hypothesis of the GW test is rejected in all cases at 1% or 5% significance level.

6. Trading Performance

Generating profitable trading signals is the main focal point of every trader. Trading profitability is not necessarily aligned with statistical accuracy. In this application, the trading performance evaluation of all models is done through a simple trading strategy. Namely, we go 'long' and 'short' when the forecasted return is positive and negative respectively. A 'long' or 'short' position means that we buy or sell respectively the ETF under study at the current price. Transaction costs severely impede the success of daily trading strategies, but ETFs offer investors the opportunity to trade stock indices with low transaction costs. In our case, the expense ratios for the seven ETFs do not exceed 0.5% per annum.[9] The out-of-sample performance of all models is presented in the following table. The trading performance measures are given in appendix C.

Table 8: Out-of-sample trading performance of every model for each ETF

Note: The table reports the annualized returns after transaction costs of every model and its respective information ratio in the parenthesis.

	ETF	**εSVR$_1$**	**εSVR$_2$**	**vSVR$_1$**	**vSVR$_2$**	**GA-εSVR**	**GA-vSVR**	**KH-LSVR$_1$**	**KH-LSVR$_2$**
F1	SPY	5.54% (1.45)	5.14% (1.36)	6.26% (1.58)	6.29% (1.61)	7.24% (1.91)	7.89% (1.97)	9.35% (2.25)	**10.27%** **(2.69)**
	QQQ	4.95% (1.37)	5.21% (1.42)	5.55% (1.55)	6.12% (1.58)	6.82% (1.84)	6.97% (1.86)	**9.84%** **(2.54)**	7.98% (2.02)
	DIA	5.95% (1.74)	6.02% (1.81)	7.11% (1.84)	8.22% (1.91)	9.05% (2.08)	9.29% (2.15)	**10.84%** **(2.89)**	9.86% (2.31)
	FEZ	6.64% (1.74)	8.73% (1.83)	9.31% (1.92)	9.26% (1.97)	9.79% (2.17)	10.07% (2.71)	10.68% (2.85)	**11.42%** **(3.11)**

[9] See, www.ishares.com/us/index

	VGK	6.05% (1.62)	6.85% (1.65)	7.39% (1.80)	8.01% (1.94)	8.54% (1.99)	9.55% (2.24)	10.35% (2.40)	**10.89% (2.87)**
	EWG	6.21% (1.74)	6.45% (1.71)	7.75% (1.88)	8.38% (1.89)	9.15% (1.99)	9.53% (2.28)	10.05% (2.68)	**10.38% (2.94)**
	EEM	7.32% (1.84)	8.31% (1.86)	9.17% (1.95)	9.44% (2.02)	9.67% (2.31)	10.80% (2.88)	10.92% (2.97)	**12.14% (3.31)**
F2	SPY	3.28% (1.29)	4.11% (1.36)	4.81% (1.58)	5.75% (1.62)	6.48% (1.74)	6.97% (1.78)	7.36% (1.83)	**8.71% (2.18)**
	QQQ	3.05% (1.22)	4.52% (1.31)	4.85% (1.41)	5.95% (1.51)	6.14% (1.53)	6.22% (1.57)	6.57% (1.64)	**7.78% (1.82)**
	DIA	3.20% (1.31)	5.18% (1.55)	5.63% (1.65)	7.26% (1.76)	7.64% (1.81)	8.24% (1.88)	8.33% (1.92)	**9.14% (2.33)**
	FEZ	7.23% (1.69)	7.18% (1.65)	8.98% (1.94)	8.90% (1.91)	9.22% (2.04)	9.59% (2.08)	**10.85% (2.87)**	10.14% (2.52)
	VGK	5.71% (1.52)	6.04% (1.69)	6.97% (1.78)	7.23% (1.89)	8.19% (1.92)	8.84% (1.98)	9.69% (2.05)	**10.04% (2.36)**
	EWG	5.78% (1.47)	5.95% (1.64)	6.28% (1.68)	7.37% (1.79)	8.40% (1.89)	8.68% (1.95)	9.17% (2.07)	**9.59% (2.23)**
	EEM	6.36% (1.61)	7.22% (1.74)	7.71% (1.82)	8.80% (1.92)	8.98% (2.14)	9.43% (2.26)	**11.67% (3.05)**	10.28% (2.94)
F3	SPY	6.48% (1.61)	6.67% (1.64)	8.86% (1.87)	9.67% (1.96)	9.28% (2.17)	9.36% (2.31)	10.05% (2.58)	**11.09% (3.14)**
	QQQ	6.34% (1.69)	6.84% (1.78)	6.87% (1.74)	6.92% (1.77)	7.15% (1.82)	7.64% (1.84)	9.81% (2.41)	**10.51% (2.74)**
	DIA	6.97% (1.71)	7.86% (1.75)	8.96% (2.08)	10.75% (2.94)	10.47% (2.87)	11.94% (3.12)	11.40% (3.24)	13.20% (3.37)
	FEZ	4.09% (1.51)	4.89% (1.58)	5.96% (1.64)	6.65% (1.75)	7.10% (1.79)	7.69% (1.85)	8.59% (1.97)	**8.92% (2.11)**
	VGK	5.14% (1.62)	5.61% (1.64)	6.24% (1.61)	7.05% (1.76)	7.54% (1.78)	8.68% (1.88)	9.27% (1.97)	**9.67%) (2.17)**
	EWG	5.41% (1.66)	5.92% (1.69)	6.49% (1.75)	7.14% (1.83)	7.84% (1.89)	8.49% (1.99)	9.14% (2.05)	**9.82% (2.29)**
	EEM	7.43% (1.92)	8.50% (1.94)	9.13% (2.08)	9.49% (2.15)	9.72% (2.28)	10.80% (2.94)	10.92% (3.06)	**13.62% (3.51)**
F4	SPY	6.18% (1.57)	6.97% (1.64)	7.96% (1.85)	9.95% (2.03)	9.87% (2.05)	10.23% (2.18)	**11.19% (3.27)**	10.64% (2.62)
	QQQ	6.52% (1.61)	6.71% (1.68)	6.96% (1.70)	7.32% (1.79)	7.48% (1.81)	7.87% (1.78)	9.77% (2.18)	**10.42% (2.67)**
	DIA	7.22% (1.73)	8.42% (1.79)	9.35% (2.07)	10.28% (2.64)	10.14% (2.52)	10.46% (2.64)	10.97% (2.88)	11.26% (3.17)
	FEZ	5.03% (1.42)	5.46% (1.58)	6.45% (1.68)	6.79% (1.71)	7.56% (1.75)	8.08% (1.80)	**9.29% (2.08)**	9.14% (1.91)
	VGK	3.68% (1.31)	4.86% (1.42)	5.84% (1.53)	6.46% (1.74)	7.51% (1.80)	8.18% (1.89)	9.38% (2.12)	**9.55% (2.20)**
	EWG	4.41% (1.34)	4.24% (1.45)	6.14% (1.62)	7.18% (1.72)	8.14% (1.82)	8.37% (1.85)	**9.35% (2.14)**	8.84% (1.94)
	EEM	6.99% (1.58)	7.66% (1.71)	8.14% (1.92)	8.95% (1.99)	9.32% (2.10)	9.67% (2.28)	10.64% (2.85)	**11.87% (3.15)**

The above table shows that KH-LSVR$_2$ delivers the best trading performance for all series and periods under study in the majority of the cases. The second best model in terms of annualized return and information ratios after transaction costs appears to be the KH-LSVR$_1$. Both KH-LSVR hybrids outperform their genetic counterparts in terms of profitability on a consistent basis. For example, during F1 on average KH-LSVRs present 1.47% higher annualized returns and 0.53 higher information ratio after transaction costs. In the same period, the KH methods outperform on average the traditional SVR techniques on average by 3.3% and 0.96 in terms of profits and information ratios. This superiority is confirmed also in the rest of the forecasting exercises. These results suggest that our trading strategy benefits from the application of the KH optimization to the locally weighted SVR process. The GA-vSVR is performing better than GA-εSVR, while both genetic approaches are found superior to the traditional SVRs in trading terms. Finally, the v-SVR techniques provide higher profits and information ratios than the ε-SVR ones. These results are consistent with the relevant literature and also with the statistical analysis performed in section 5. Table 9 below examines the average trading performances of all models for all forecasting exercises.

Table 9: Average trading performances per forecasting exercise

Note: The table reports the average annualized returns after transaction costs of all models. Their information ratios are presented in the parenthesis. The total average corresponds to the average trading results over all ETFs. US and EU averages refer to the average trading performance of all models over SPY, QQQ, DIA and FEZ, VGK, EWG respectively.

	F1	**F2**	**F3**	**F4**
Total Average	8.34% (2.09)	7.39% (1.85)	8.33% (2.10)	8.17% (1.99)
US Average	7.41% (1.91)	6.13% (1.65)	8.96% (2.26)	8.92% (2.16)
EU Average	8.81% (2.16)	8.17% (1.94)	7.12% (1.82)	7.08% (1.74)
EEM Average	9.72% (2.39)	8.81% (2.19)	9.95% (2.49)	9.16% (2.20)

As it turns out, the total average trading performance is worse in F2. This is expected since the peak of the global financial crisis is included in the out-of-sample period of F2. ETFs tracking the performance of US market are found to be less and more profitable in 2008 (F1, F2) and 2012 (F3, F4) respectively. This is also not surprising because the US economy was affected more by the global financial crisis than the European sovereign debt one. The trend is opposite when it comes to European ETFs. Tables 8 and 9 results suggest that the profitability of the emerging market ETF is higher overall.

The outcomes of the above trading exercises show that all models generate profits even in volatile economic periods, while complex techniques prove to be more successful than the simple ones. It is also worth noting that the trading performance of all models seems to vary through time. These two observations are consistent with the AMH. The hypothesis states that the performance of trading models varies through time and it deteriorates in times of market turbulence.

In the next section, we decompose the trading performance of our models in the out-of-sample periods. This will allow us to test one further main implication of the AMH. The hypothesis that the profitability of all models is diminishing through time (else trading models are not robust in the long run). Additionally it will be interesting to see the “diminishing rate” in the different periods. AMH implies that the rate should be higher when the market is in crisis. Table 10 presents the monthly trading performance of all models and periods for the most profitable ETFs tracking the US and EU markets, along with the one tracking the performance of emerging markets ETF[10].

[10] The results of the remaining four ETFs are not presented here for the sake of space. The pattern of their returns is similar with those presented in Tables 10.

Table 10: Monthly Out-of-sample Trading Performance of selected ETFs

Note: The table reports the monthly annualized returns after transaction costs of every model. DIA and FEZ cases are presented as they are the most profitable US and EU ETFs (on average).

		F1 (Out-of-sample)						F2 (Out-of-sample)					
ETF	**Models**	**Jan**	**Feb**	**Mar**	**Apr**	**May**	**Jun**	**Jul**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
DIA	εSVR$_1$	8.97%	6.13%	7.32%	5.70%	4.70%	2.87%	6.64%	2.85%	2.74%	2.18%	2.45%	2.31%
	εSVR$_2$	8.14%	8.24%	6.58%	5.94%	4.14%	3.10%	8.21%	6.12%	5.87%	3.44%	3.89%	3.54%
	vSVR$_1$	9.77%	8.49%	7.48%	7.13%	5.64%	4.16%	7.97%	7.52%	5.84%	4.28%	4.17%	3.98%
	vSVR$_2$	10.02%	9.14%	8.88%	7.51%	7.08%	6.70%	9.24%	8.07%	7.75%	6.42%	6.18%	5.87%
	GA-εSVR	10.85%	11.07%	8.34%	8.73%	7.81%	7.52%	10.62%	9.69%	7.64%	6.87%	5.88%	5.14%
	GA-vSVR	11.76%	10.45%	9.42%	8.84%	7.74%	7.52%	11.05%	10.60%	8.10%	7.08%	7.67%	4.96%
	KH-LSVR$_1$	12.93%	11.88%	10.84%	10.26%	9.85%	9.28%	11.48%	10.64%	9.37%	8.72%	5.58%	4.20%
	KH-LSVR$_2$	12.45%	11.27%	10.78%	7.92%	8.56%	8.20%	10.97%	11.08%	10.42%	10.24%	6.38%	5.72%
FEZ	εSVR$_1$	8.12%	7.05%	6.85%	6.44%	6.13%	5.24%	7.69%	8.15%	7.93%	7.65%	6.94%	5.01%
	εSVR$_2$	10.04%	10.58%	9.92%	8.58%	7.90%	5.34%	9.50%	7.82%	7.54%	6.87%	6.25%	5.11%
	vSVR$_1$	11.22%	11.00%	10.24%	9.52%	8.12%	5.76%	10.55%	10.23%	9.85%	8.94%	8.17%	6.13%
	vSVR$_2$	12.08%	10.25%	9.84%	9.37%	7.78%	6.26%	10.42%	10.62%	9.84%	8.96%	7.34%	6.22%
	GA-εSVR	12.84%	11.80%	10.03%	9.81%	7.64%	6.61%	11.38%	11.02%	10.05%	8.40%	7.94%	6.50%
	GA-vSVR	12.15%	12.43%	10.81%	9.65%	8.14%	7.21%	11.88%	11.64%	10.15%	9.63%	8.54%	5.72%
	KH-LSVR$_1$	12.84%	11.82%	10.83%	9.66%	9.75%	9.18%	12.24%	12.38%	11.45%	10.53%	9.52%	8.95%
	KH-LSVR$_2$	13.42%	12.67%	11.68%	10.05%	10.23%	10.47%	12.52%	11.71%	10.38%	9.88%	8.59%	7.74%
EEM	εSVR$_1$	9.25%	8.56%	8.15%	7.86%	5.22%	4.89%	8.15%	7.82%	6.95%	6.12%	5.45%	3.64%
	εSVR$_2$	9.82%	9.86%	8.86%	8.65%	7.11%	5.54%	8.67%	8.35%	7.68%	7.21%	6.38%	5.04%
	vSVR$_1$	10.52%	10.58%	10.03%	9.52%	8.25%	6.11%	9.80%	9.39%	9.20%	7.98%	5.22%	4.68%
	vSVR$_2$	10.95%	10.23%	9.68%	9.27%	8.93%	7.55%	10.62%	10.11%	8.79%	8.14%	7.98%	7.14%
	GA-εSVR	11.16%	10.93%	9.40%	9.46%	9.10%	7.95%	10.75%	10.09%	9.87%	8.84%	8.16%	6.15%
	GA-vSVR	13.61%	12.18%	10.38%	9.95%	9.67%	8.99%	12.08%	11.00%	9.41%	9.24%	7.65%	7.18%
	KH-LSVR$_1$	12.74%	12.26%	11.45%	10.44%	10.42%	8.18%	13.74%	12.88%	12.17%	11.02%	10.25%	9.95%
	KH-LSVR$_2$	14.28%	13.14%	13.56%	10.88%	10.70%	10.28%	12.24%	11.84%	10.57%	9.48%	9.06%	8.51%
		F3 (Out-of-sample)						**F4 (Out-of-sample)**					
ETF	**Models**	**Jan**	**Feb**	**Mar**	**Apr**	**May**	**Jun**	**Jul**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
DIA	**εSVR$_1$**	8.65%	7.45%	6.97%	6.57%	6.10%	6.08%	8.42%	8.12%	7.68%	6.64%	6.49%	5.94%
	εSVR$_2$	9.41%	8.68%	7.80%	7.62%	6.84%	6.78%	9.22%	9.53%	9.12%	8.84%	7.16%	6.62%
	vSVR$_1$	10.65%	9.65%	9.24%	8.84%	7.86%	7.54%	10.63%	10.24%	10.75%	8.74%	8.08%	7.66%
	vSVR$_2$	12.32%	11.84%	10.70%	10.56%	9.58%	9.47%	11.95%	11.41%	10.32%	10.14%	9.42%	8.44%
	GA-εSVR	12.61%	12.72%	11.43%	10.24%	9.27%	6.54%	12.14%	12.05%	11.21%	9.94%	9.17%	6.35%
	GA-vSVR	13.42%	12.86%	12.83%	11.96%	10.68%	9.86%	12.35%	12.02%	10.15%	10.08%	9.22%	8.95%
	KH-LSVR$_1$	13.74%	13.05%	11.66%	10.42%	10.18%	9.35%	13.55%	12.42%	11.24%	10.10%	9.34%	9.17%
	KH-LSVR$_2$	14.81%	14.54%	13.70%	12.97%	12.65%	10.51%	13.69%	12.28%	11.24%	10.25%	10.16%	9.92%
FEZ	**εSVR$_1$**	8.32%	5.33%	4.15%	3.18%	2.01%	1.54%	7.87%	6.14%	5.45%	4.39%	4.16%	2.15%
	εSVR$_2$	8.85%	6.42%	4.64%	3.86%	3.45%	2.11%	8.74%	6.75%	5.38%	4.64%	3.95%	3.32%
	vSVR$_1$	8.98%	7.84%	6.15%	5.06%	4.61%	3.10%	9.78%	8.68%	7.22%	5.62%	3.85%	3.56%
	vSVR$_2$	9.65%	8.22%	7.03%	6.24%	5.40%	3.37%	9.78%	8.51%	7.15%	6.40%	4.66%	4.24%
	GA-εSVR	9.74%	9.14%	7.64%	6.61%	5.22%	4.23%	10.06%	9.25%	8.07%	7.23%	5.86%	4.89%
	GA-vSVR	10.62%	9.53%	8.41%	6.78%	5.69%	5.08%	10.73%	9.94%	8.48%	7.56%	6.40%	5.38%
	KH-LSVR$_1$	10.96%	10.48%	9.46%	7.66%	6.61%	6.37%	11.24%	10.17%	9.80%	8.42%	8.25%	7.84%
	KH-LSVR$_2$	11.64%	9.96%	9.71%	8.65%	6.95%	6.62%	10.86%	10.55%	9.64%	8.84%	8.32%	6.63%
EEM	**εSVR$_1$**	9.39%	9.73%	7.89%	7.03%	6.16%	4.35%	9.15%	9.28%	7.56%	6.20%	5.23%	4.49%
	εSVR$_2$	10.75%	9.84%	8.45%	8.14%	7.23%	6.57%	9.48%	9.18%	7.84%	7.59%	6.48%	5.36%
	vSVR$_1$	10.42%	10.09%	9.82%	8.94%	8.04%	7.47%	9.94%	9.49%	9.21%	8.45%	6.18%	5.56%
	vSVR$_2$	10.48%	10.20%	9.92%	9.24%	8.95%	8.12%	10.35%	10.17%	9.84%	8.75%	7.78%	6.82%
	GA-εSVR	11.42%	10.53%	10.56%	9.60%	8.34%	7.84%	10.84%	10.41%	9.81%	9.32%	8.25%	7.31%
	GA-vSVR	13.45%	11.75%	10.86%	10.06%	9.38%	9.31%	11.28%	10.49%	9.97%	9.38%	9.15%	7.74%
	KH-LSVR$_1$	13.86%	12.34%	10.97%	10.17%	9.43%	8.76%	12.84%	10.85%	10.62%	10.48%	10.31%	8.71%
	KH-LSVR$_2$	14.85%	13.97%	13.84%	13.24%	13.16%	12.65%	13.71%	13.07%	12.48%	11.26%	10.74%	9.94%

The above results show that the profitability of all models is declining as time passes by. The declining speed seems to be increased after the third month of the out-of-sample. In Table 11, this decay rate (estimated as: (current month average return-previous month average return)/ previous month average return) along with its average for each forecasting exercise is presented.

Table 11: Monthly Returns Decay Speed

*Note: Columns (1)-(5) present the decay rate for the second to the sixth month of the out-of-sample. The last column is the average while in the parenthesis is the p-value of the test of equal means between the current the current exercise and the previous one for each ETF. *** and ** denote a rejection of the null hypothesis at the 1% and 5% significance level respectively.*

ETF	Forecasting Exercise	(1)	(2)	(3)	(4)	(5)	Average
DIA	F1	-9.68%	-9.17%	-10.93%	-10.49%	-11.11%	-10.28%
	F2	-12.61%	-13.28%	-14.72%	-14.28%	-15.36%	-14.05% (0.000***)
	F3	-5.04%	-7.12%	-6.11%	-7.60%	-9.61%	-7.10% (0.000***)
	F4	-4.22%	-7.22%	-8.54%	-7.61%	-8.68%	-7.25% (0.889)
FEZ	F1	-5.51%	-8.45%	-8.88%	-10.11%	-14.64%	-9.52%
	F2	-3.03%	-7.63%	-8.20%	-10.68%	-18.82%	-9.67% (0.960)
	F3	-15.03%	-14.54%	-16.00%	-16.86%	-18.83%	-16.25% (0.041**)
	F4	-11.47%	-12.57%	-13.22%	-14.41%	-16.37%	-13.61% (0.047**)
EEM	F1	-4.97%	-7.10%	-6.72%	-8.72%	-14.28%	-8.36%
	F2	-5.31%	-8.39%	-8.86%	-11.58%	-13.07%	-9.44% (0.618)
	F3	-6.52%	-6.94%	-7.16%	-7.50%	-7.95%	-7.21% (0.142)
	F4	-5.31%	-6.76%	-7.63%	-10.23%	-12.77%	-8.54% (0.354)

Table 11 shows that decay rate is increasing through time. It is worth noting that on average the decay rate for DIA is higher on F2 (the peak of the global financial crisis) and for FEZ during F3 (when the European sovereign debt crisis reached its maximum). The p-values suggest that the decay rates for DIA are statistical different between F1, F2 and F2, F3. It seems that the decay rate during the global financial crisis is statistically significant higher during F2. A similar view can be obtained for FEZ and period F3. The results for the emerging markets index (EEM) are statistically insignificant. Emerging markets were affected during both crises however not to the extent of advanced ones. These results allow us to conclude that for the periods under study our models are less robust when the underlying forecasted market is in crisis. Under normal or near to normal market conditions the decay rates seem similar.

7. Conclusions

In this research a hybrid KH-LSVR model is introduced. The KH algorithm is a novel metaheuristic optimization technique inspired by the behaviour of krill herds. The KH is used to optimize the LSVR parameters by balancing the search between local and global optima. The proposed model is evaluated through three different fitness functions, while its statistical and trading performance is benchmarked against seven traditional SVR structures. The inputs of the SVR models are selected through a novel statistical technique that involves a large pool of linear and non-linear predictors. The technique incorporates PCA analysis and the MCS test proposed by Hansen *et al.* (2011). All models are applied in four forecasting and trading exercises over seven ETFs during the period 2006-2012. The purpose of the trading applications is to test the robustness of the models under study and to provide empirical evidence in favour of the AMH.

In terms of the results, KH-LSVR architectures outperform their counterparts in terms of statistical accuracy and trading efficiency. The trading application provides evidences in favour of the AMH, while the SVR input selection process seems successful. This work should go forward on convincing researchers, practitioners and academics to explore further hybrid SVR techniques. It should also serve as caution on the implications of the AMH and the robustness of trading models.

References

Aguilar-Rivera, R., Valenzuela-Rendón, M. & Rodríguez-Ortiz, J.J. (2015) Genetic algorithms and Darwinian approaches in financial applications: A survey, Expert Systems with Applications, 42 (21), pp. 7684-7697.

Bernstein, F., Li, Y. & Shang, K. (2015) A simple heuristic for joint inventory and pricing models with lead time and backorders. Management Science. Forthcoming.

Chan, K.S.& Tong, H. (1986) On estimating thresholds in autoregressive models, Journal of Time Series Analysis, 7 (3), pp. 178–190.

Chang, T.J., Meade, N., Beasley, J.E. & Sharaiha, Y.M. (2000) Heuristics for cardinality constrained portfolio optimization, Computers and Operations Research, 27 (13), pp. 1271–1302.

Cherkassky,V. & Ma,Y. (2004) Practical selection of SVM parameters and noise estimation for SVM regression, Neural Networks, 17 (1),pp. 113-126.

Cleveland, W.S. & Devlin, S.J. (1988) Locally Weighted Regression: an approach to regression analysis by ocal fitting, Journal of the American Statistical Association, 83 (403), pp. 596-610.

Diebold, F.X. & Mariano, R.S. (1995) Comparing predictive ability, Journal of Business & Economic Statistics, 13 (3), pp. 253-263

Dolvin, D. (2010) S&P ETFs: arbitrage opportunities and market forecasting, The Journal of Index Investing (2010), 1(1), pp. 107–116.

Dunis, C.L., Laws, J. & Sermpinis, G. (2011) Higher order and recurrent neural architectures for trading the EUR/USD exchange rate, Quantitative Finance, 11 (4), pp. 615-629.

Dunis, C.L. & Nathani, A. (2007) Quantitative trading of gold and silver using non-linear models, Neural Network World, 16 (2), pp. 93-111.

Elman, J.L. (1990) Finding structure in time, Cognitive Science, 14 (2), pp. 179-211.

Ferreira, C. (2001) Gene Expression Programming: A New Adaptive Algorithm for Solving Problems. Complex Systems, 13(2), pp. 87-129.

Ferri, R.A. (2009) The ETF Book: All you need to know about Exchange-Traded Funds, John Wiley & Sons: Hoboken, New Jersey.

Fix, E. & Hodges, J.L. (1951) Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas.

Gandomi, A.H. & Alavi, A.H. (2012) Krill herd: A new bio-inspired optimization algorithm, Communications in Nonlinear Science and Numerical Simulation, 17 (12), pp. 4831-4845.

Giacomini, R. & White, H. (2006) Tests of conditional predictive ability, Econometrica, 74(6), pp. 1545–1578.

Gilli, M., Maringer, D & Winker, P. (2008) Applications of heuristics in finance. In: Handbook On Information Technology in Finance, Seese, D., Weinhardt,C. & Schlottman, F. (Eds.), International Handbooks on Information Systems. Springer, Germany, pp. 635–653.

Goldberg, D.E. (1989) Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley.

Ghosh, J. & Shin, Y. (1991) The Pi-Sigma Network: An efficient higher-order neural networks for pattern classification and function approximation, Proceedings of International Joint Conference of Neural Networks, 1, pp. 13-18.

Hall, N.G. & Posner, M.E. (2007) Performance prediction and preselection for optimization and heuristic solution procedures. Operations research,55(4), pp.703-716.

Hansen, P.R, Lunde, A. & Nason, J.M. (2011) The Model Confidence Set, Econometrica, 79 (2), pp. 453-497.

Holland, J. (1975) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control and artificial intelligence, Cambridge, Mass: MIT Press.

Hsu, S.H., Hsieh, J.J.P.A., Chih, T.C. & Hsu, K.C. (2009) A two-stage architecture for stock price forecasting by integrating self-organizing map and support vector regression, Expert Systems with Applications, 36 (4), pp. 7947-7951.

Huang, K., Yang, H., King, I. & Lyu, M. R. (2006) Local support vector regression for financial time series prediction, Proceedings of International Joint Conference on Neural Networks, pp. 1622-1627.

Huang, S.C., Chuang, P.J., Wu, C.F. & Lai, H.J. (2010) Chaos-based support vector regressions for exchange rate forecasting, Expert Systems with Applications, 37(12),pp. 8590-8598.

Jarque, C.M. & Bera, A.K. (1980) Efficient tests for normality, homoscedasticity and serial independence of regression residuals, Economics Letters, 6 (3), pp. 255-259.

Jiang, H. & He, W. (2012) Grey relational grade in local support vector regression for financial time series prediction, Expert Systems with Applications, 39 (3), pp. 2256-2262.

Kao,L.J.,Chiu,C.C.,Lu,C.J., Yang, J.L. (2013) Integration of nonlinear independent component analysis and support vector regression for stock price forecasting, Neurocomputing, 99(1), pp. 534-542.

Karaboga, D. & Basturk, B. (2008) On the performance of artificial bee colony (ABC) algorithm, Applied Soft Computing, 8 (1), pp. 687-697.

Koza, J., & Poli, R., (2005) Genetic Programming, in: Burke, E.K., Kendall, G. (eds.), Search Methodologies, Introductory Tutorials in Optimization and Decision Support Techniques, Springer, pp.127-164.

Lee, DE., Song, JH., Song, SO., Yoon, ES. (2005), Weighted support vector machine for quality estimation in the polymerization process, Industrial & Engineering Chemistry Research, 44(7), pp. 2101-2105.

Li, X., Zhang, J., Yin, M. (2014) Animal migration optimization: an optimization algorithm inspired by animal migration behavior, Neural Computing and Applications, 24 (7-8), pp. 1867-1877.

Liang, J.J., Qin, A.K., Suganthan, P.N. & Baskar, S. (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Transactions on Evolutionary Computations, 10 (3), pp. 281–295.

Lin, K.P. & Pai, P.F. (2010) A fuzzy support vector regression model for business cycle predictions, Expert Systems with Applications, 37 (7), pp. 5430-5435.

Lin, C.J. & Teräsvirta, T. (1994) Testing the constancy of regression parameters against continuous structural changes, Journal of Econometrics, 62 (2), pp. 211–228.

Lo, A. W. (2000). Finance: A selective survey. Journal of the American Statistical Association, 95 (450), 629-635.

Lo, A.W. (2004) The adaptive markets hypothesis, Journal of Portfolio Management, 30, pp. 15–29.

Lu, C.J., Lee, T.S., Chiu, C.C. (2009) Financial time series forecasting using independent component analysis and support vector regression, Decision Support Systems, 47 (2), pp. 115-125.

Martí, R., Gallego, M., Abraham Duarte, M. & Pardo, E.G. (2013) Heuristics and metaheuristics for the maximum diversity problem, Journal of Heuristics , 19 (4), pp. 591-615.

Marshall, B.R., Nguyen, N.H & Visaltanachoti, N. (2013) ETF arbitrage: Intraday evidence, Journal of Banking & Finance, 37 (9), pp. 3486-3498.

Oreski, S. & Oreski, G. (2014) Genetic algorithm-based heuristic for feature selection in credit risk assessment, Expert Systems with Applications, 41 (4), pp. 2052-2064.

Pai, P.F., Lin, C.S., Hong, W.C. & Chen, C.T. (2006) A hybrid support vector machine regression for exchange rate prediction, International Journal of Information and Management Sciences, 17 (2), pp. 19-32.

Park, C.H. and Irwin, S.H. (2007) What do we know about profitability of technical analysis?, Journal of Economic Surveys, 21 (4), pp. 786–826.

Pesaran, M.H. & Timmerman, A.G. (1992) A simple nonparametric test of predictive performance, Journal of Business and Economic Statistics, 10(4), pp. 461-465.

Popescu, D. & Crama, P. (2015) Advertising revenue optimization in live television broadcasting. Management Science, Forthcoming.

Ruiz, A. & Lopez-de-Teruel, P.E. (2001) Nonlinear kernel-based statistical pattern analysis, IEEE Transactions on Neural Networks, 12 (1), pp. 16-32.

Schölkopf, B., Bartlett, P., Smola, A., Williamson, R. (1999), Shrinking the tube: a new support vector regression algorithm, In: Kearns, M.J., (ed.), Advances in neural information processing systems 11. Cambridge, Mass, MIT Press, pp. 330-336.

Shapiro, A.F. (2000) A Hitchhiker's guide to the techniques of adaptive non-linear models. Insurance: Mathematics and Economics, 26 (2-3), pp. 119-132.
Sermpinis, G., Theofilatos, K.A, Karathanasopoulos, A.S., Georgopoulos, E.F. & Dunis, C.L. (2013) Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and Particle Swarm Optimization, European Journal of Operational Research, 225 (3), pp. 528–540.
Sermpinis, G., Stasinakis, C., Theofilatos, K. & Karathansopoulos, A. (2015) Modeling, forecasting and trading the EUR exchange rates with hybrid rolling genetic algorithms – support vector regression forecast combinations, Journal of Operational Research, 247(3), pp. 831-846.
Suykens, J.A.K., De Brabanter, J., Lukas, L. & Vandewalle, J. (2002) Weighted least squares support vector machines: robustness and sparse approximation, Neurocomputing, 48 (1-4), pp. 85-105.
Talbi, E.G. (2009) Metaheuristics: from design to implementation. Hoboken, New Jersey, USA: John Wiley & Sons.
Tenti, P. (1996) Forecasting foreign exchange rates using recurrent neural networks, Applied Artificial Intelligence, 10 (6), pp. 567-582.
Timmermann, A. (2006) Forecast Combinations, In: Handbook of Economic Forecasting, Elliott, G. Granger, C.W.J. &. Timmermann, A. (Eds), 1, pp. 135-196.
Trafalis, T.B., Ince, H. (2000) Support vector machine for regression and applications to financial forecasting, IJCNN, Neural Networks, IEEE, 6, pp. 348-353.
Turmon, M. J. (1998) Machine Learning and Statistics: The Interface. Journal of the American Statistical Association, 93 (442), pp. 833-835.
Vapnik, V. (1995) The nature of statistic learning theory, Springer-Verlag, New York.
Wagner, D. (2011) Trading ETFs: Gaining an edge with technical analysis, John Wiley & Sons: Hoboken, New Jersey.
Wang,G.G., Guo,L., Gandomi,A.H., Hao,G.S. & Wang,H. (2014) Chaotic Krill Herd algorithm, Information Sciences, 274 (1), pp. 17-34.
Wu, S. & Akbarov, A. (2011) Support vector regression for warranty claim forecasting, European Journal of Operational Research, 213 (1), pp. 196-204.
Wu, CH., Tzeng, GH., Goo, JY. & Fang, WC. (2007) A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy, Expert Systems with Applications, 32 (2), pp. 397-408.
Wu, Y. & Liu, Y. (2007) Robust truncated hinge loss support vector machines, Journal of the American Statistical Association, 102(479), pp. 974-983.
Yang, H., Huang, K., King, I. & Lyu, M.R. (2009) Localized support vector regression for time series prediction, Neurocomputing, 72 (10–12), pp. 2659-2669.
Yang, X.S. (2010) Firefly algorithm nature-inspired metaheuristic algorithms. United Kingdom: Luniver Press.
Yang, X.S. & Gandomi, A.H. (2012) Bat algorithm: a novel approach for global engineering optimization, Engineering Computations, 29 (5), pp. 464–483.
Yeh,CY., Huang,CW. & Lee, SJ. (2011), A multiple-kernel support vector regression approach for stock market price forecasting, Expert Systems with Applications, (3), pp.2177-2186.
Yuan, FC. (2012) Parameters optimization using genetic algorithms in support vector regression for sales volume forecasting, Applied Mathematics, 3 (1), pp. 1480-1486.
Zhang, G., Hu,M.Y., Patuwo, B.E. & Indro, D.C. (1999) Artificial neural networks in bankruptcy prediction: General framework and cross-validation analysis, European Journal of Operational Research, 116 (1), pp. 16-32.
Zhang, L., Zhou, W. & Jiao, L. (2004) Wavelet support vector machine, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 34 (1), 34-39.

Appendix

A. Predictors' pool

This appendix section is a short description of the linear and non-linear models used to populate the individual forecast pools.

A.1 Linear Predictors

The linear models used are SMA, EMA, AR, ARMA, ROC and PPI. Their specifications are provided in the following table. In total, the linear models' forecasts sum up to 300.

Table A.1: The specification of the linear models

LINEAR MODELS	DESCRIPTION	TOTAL INDIVIDUAL FORECASTS
SMA (*q*)	$E(R_t) = (R_{t-1} + ... + R_{t-q}) / q$ *Where:* • *q=3...25*	23
EMA (*q'*)	$E(R_t) = \frac{R_{t-1} + (1-a')R_{t-2} + ... + (1-a')^{q'-1} R_{t-q'}}{a' + (1-a') + ... + (1-a')^{q'-1}}$ *Where:* • *q'=3...25* • *a'*=2/(1+N*days*), *Ndays is the number trading days*	23
AR (*q''*)	$E(R_t) = \beta_0 + \sum_{i'=1}^{q''} \beta_{i'} R_{t-i'}$ *Where:* • *q''=1,…,20* • $\beta_0, \beta_{i'}$ *the regression coefficients*	20
ARMA (*m'*, *n'*)	$E(R_t) = \bar{\varphi}_0 + \sum_{j'=1}^{m'} \bar{\varphi}_{j'} R_{t-j'} + \bar{a}_0 + \sum_{k'=1}^{n'} \bar{w}_{k'} \bar{a}_{t-k'}$ *Where:* • *m', n'=1,..,15* • $\bar{\varphi}_0, \bar{\varphi}_{j'}$ *the regression coefficients* • $\bar{a}_0, \bar{a}_{t-k'}$ *the residual terms* • $\bar{w}_{k'}$ *the weights of the residual terms*	210
ROC (*p'*)	$E(R_t) = 100[1 - (R_{t-1} / R_{t-p'})]$ *Where:* • *p'=3,..,25*	23
PPI	$\begin{Bmatrix} PivotP_{t-1} = (H_{t-1} + L_{t-1} + CP_{t-1}) / 3 \\ E(R_t) = (PivotP_{t-1} - CP_{t-1}) / CP_{t-1} \end{Bmatrix}$ *Where:* • $PivotP_{t-1}$ *the pivot point for t-1* • $H_{t-1}, L_{t-1}, CP_{t-1}$ *the high, low and closing price for t-1*	1

A.2 Non-linear Predictors

A.2.1. Smooth Transition Autoregressive Models (STAR)

STAR as proposed by Chan and Tong (1986) are extensions of the ARs. The STAR combines two AR models with a function that defines the degree of non-linearity (smooth transition function).

$$E(R_t) = \Phi_1' X_t (1 - F'(z_t', \zeta', \lambda')) + \Phi_2' X_t F'(z_t', \zeta', \lambda') + u_t' \quad \text{(B.1)}$$

Where: $\Phi_i' = (\tilde{\varphi}_{i,0}, \tilde{\varphi}_{i,1}, \ldots \tilde{\varphi}_{i,p}),\ i = 1, 2$ and $\tilde{\varphi}_{i,0}, \tilde{\varphi}_{i,1}, \ldots \tilde{\varphi}_{i,p}$ the regression coefficients of the two AR models, $X_t = (1, \tilde{\chi}_t')'$ with $\tilde{\chi}_t' = (R_{t-1}, \ldots, R_{t-p''})$, $0 \le F'(z'_t, \zeta', \lambda') \le 1$ the smooth transition function, $z'_t = R_{t-d'}, d' > 0$ the lagged endogenous transition variable, ζ' the parameter that defines the smoothness of the transition between the two regimes, λ' the threshold parameter and $u_{t'}$ the error term. In this case, we estimate two-regime logistic (LSTAR) and exponential (ESTAR) STARs (Lin and Teräsvirta, 1994). For both models the orders 1 to 20 are explored.

A.2.2. Nearest Neighbors Algorithm (*k*-NN)

Nearest Neighbors (*k*-NN) is a non-linear and non-parametric forecasting method based on the work of Fix and Hodges (1951). It is based on the idea that pieces of time series in the past have patterns which might have resemblance to pieces in the future. Similar patterns of behavior are located in terms of nearest neighbors using a distance called the Euclidean distance and these patterns are used to predict behavior in the immediate future. We follow the guidelines of Dunis and Nathani (2007) to define the parameters. The optimal set of parameters is selected based on the highest trading performance in the in-sample period.

A.2.3. Neural Networks (NNs)

In this analysis, five NN architectures are applied. The simpler and most popular is the MLP. A standard MLP has at least three layers. The first layer is called the input layer (the number of its nodes corresponds to the number of explanatory variables). The last layer is called the output layer (the number of its nodes corresponds to the number of response variables). An intermediary layer of nodes, the hidden layer, separates the input from the output layer. Its number of nodes defines the amount of complexity the model is capable of fitting. In addition, the input and hidden layer contain an extra node called the bias node. This node has a fixed value of one and has the same function as the intercept in traditional regression models. Normally, each node of one layer has connections to all the other nodes of the next layer.

The training of the network (which is the adjustment of its weights in a way that the network maps the input value of the training data to the corresponding output value) starts with randomly chosen weights and proceeds by applying a learning algorithm called back-propagation of errors (Shapiro, 2000). The maximum number of the allowed back-propagation iterations is optimized by maximizing a fitness function in the test dataset (see table 2) through a trial and error procedure. More specifically, the learning algorithm tries to find those weights which minimize an error function (normally the sum of all squared differences between target and actual values). Since networks with sufficient hidden nodes are able to learn the training data (as well as their outliers and their noise) by heart, it is crucial to stop the training procedure at the right time to prevent overfitting (this is called 'early stopping'). This is achieved by dividing the dataset into 3 subsets respectively called the training and test sets used for simulating the data currently available to fit and tune the model and the validation set used for simulating future values. The network parameters are then estimated by fitting the training data using the backpropagation of errors. The iteration length is optimized by maximizing the forecasting accuracy for the test dataset. Then the predictive value of the model is evaluated applying it to the validation dataset (out-of-sample dataset).

A Recurrent Neural Network is also applied. For an exact specification of recurrent networks, see Elman (1990). A simple recurrent network has an activation feedback which embodies short-term memory. In other words, the RNN architecture can provide more accurate outputs because the inputs are (potentially) taken from all previous values. Although RNN require substantially more computational time (see Tenti (1996), they can yield better results in comparison with simple MLPs due to the additional memory inputs. The third NN model included in the feature space is the Higher Order Neural Network (HONN). HONNs are able to simulate higher frequency, higher order non-linear data, and consequently provide superior simulations. For more information on HONNs see Dunis *et al.* (2011). Psi Sigma Networks (PSNs) are considered as a class of feed-forward fully connected HONNs. First introduced by Ghosh and Shin (1991), the PSN creation was motivated by the need to create a network combining the fast learning property of single layer networks with the powerful mapping capability of HONNs, while avoiding the combinatorial increase in the required number of weights. The order of the network in the context of PSN is represented by the number of hidden nodes. In a PSN the weights from the hidden to the output layer are fixed to 1 and only the weights from the input to the hidden layer are adjusted, something that greatly reduces the training time.

The last NN used is the ARBF-PSO. Its complexity, architecture and characteristics differ from the previous mentioned NNs. Compared to them, in the ARBF-PSO the parameters are optimized through a Particle Swarm Optimization (PSO)[11] algorithm. This protects the ARBF-PSO from the dangers of over-fitting and data snooping. However, the practitioner still needs to select the network's inputs as with the previous NNs. For a complete description of the ARBF-PSO see Sermpinis *et al.* (2013).

[11] The PSO algorithm is a population based heuristic search algorithm based on the social behavior of birds within a flock (Liang et al, 2006).

Table B.2 summarizes the learning algorithm, hidden and output node activation functions for all previous structures.

Table A.2: Neural Network Design and Training Characteristics

Note: The input of every node is z_ψ, where $\psi = 1 \dots n''$ and n'' is the number of nodes of the previous layer. The vector indicating the center of the Gaussian function is C' and σ' is the value indicating its width.

PARAMETERS	MLP	RNN	HONN	PSN	ARBF-PSO
Learning algorithm	Gradient descent	Gradient descent	Gradient descent	Gradient descent	Particle Swarm Optimization
Initialisation of weights	N(0,1)	N(0,1)	N(0,1)	N(0,1)	-
Hidden node activation function	$F(z_\psi)=1/(1+e^{-z_\psi})$	$F(z_\psi)=1/(1+e^{-z_\psi})$	$F(z_\psi)=1/(1+e^{-z_\psi})$	$F(z_\psi)=\sum_{\psi=1}^{n''} z_\psi$	$F(z_\psi)=\exp\left(\frac{\lVert z_\psi - C'\rVert^2}{2\sigma'^2}\right)$
Output node activation function	$F(z_\psi)=\sum_{\psi=1}^{n''} z_\psi$	$F(z_\psi)=\sum_{\psi=1}^{n''} z_\psi$	$F(z_\psi)=\sum_{\psi=1}^{n''} z_\psi$	$F(z_\psi)=1/(1+e^{-z_\psi})$	$F(z_\psi)=\sum_{\psi=1}^{n''} z_\psi$

There is no formal theory behind the selection of the NN inputs and their characteristics, such as number of hidden neurons, learning rate, momentum and iterations. We conduct NN experiments and a sensitivity analysis on a pool of autoregressive and autoregressive-moving average terms of all series in the in-sample dataset. For example for the number of iterations, our experimentation started from 1.000 iterations and stopped at the 100.000 iterations, increasing in each experiment the number of iterations by 1.000. This is a very common approach in the literature (Tenti, 1996). Based on these experiments and the sensitivity analysis, the sets of variables selected are those that provide the higher trading performance for each network in the in-sample period.

A.2.4. Genetic Programming predictors

The final two non-linear models are the GP and GEP. These techniques are domain-independent problem solving methods based on the Darwinian principle of reproduction and survival of the fittest. GP, as class of GAs, creates an initial population of models and evolves it using genetic operators (crossover and mutation). The result is to perform mathematical expressions that best fit to the given input (data). When designing a GP algorithm the main focus is on optimizing execution time and limiting the 'bloat effect', a similar issue to over-fitting in NNs mentioned earlier. This genetic procedure creates superior offsprings, replacing the worst models (tournament losers), and rearranges the initial population for the next iteration. The iterations stop and the final forecast results are obtained when the model reaches the critical value of the termination criterion. GP holds a greater selection strength and genetic drift from a typical GA. The functionality aspects of GP and the genetic operators are described in detail by Koza and Poli (2005).

GEP is based on symbolic strings of fixed length that represent the genotype of an organism. Its chromosomes consist of multiple genes with equal lengths. Each gene includes a head (detailing symbols specific to functions and terminals) and a tail (only includes terminals). The set of terminals included within both the heads and tails of the chromosomes comprises constants and specific variables. Each gene holds the capacity to code for multiple and different expression trees. Valid expression trees are always generated when using GEP, while it can operate also when the first element of a gene is terminal. This is not guaranteed in GP. GEP is also able to code for sub-expression trees with interlinking functions in order to enable reproduction when multiple generations arise. In general, GEP is considered superior to GP because fitness is established through the genotype and phenotype of an individual based on its chromosomes and expression trees respectively. Ferreira (2001) provides details on the exact procedure of GEP.

B. Parameters and training characteristics

The table B.1 summarizes the training characteristics of the GA and KH algorithm for all ETFs and forecasting exercises.

Table B.1: GA and KH training characteristics

	GA					**KH**				
	Forecasting Exercise	1	2	3	4	**Forecasting Exercise**	1	2	3	4
SPY	**Population Size**	60	60	60	60	**Population Size**	60	60	60	60
	Maximum Generations	1000	1000	1000	1000	Δt , Z_{cr}	20.12, 0.39	20.12, 0.39	20.12, 0.39	20.12, 0.39
QQQ	**Population Size**	45	45	45	45	**Population Size**	45	45	45	45
	Maximum Generations	850	850	850	850	Δt , Z_{cr}	21.28, 0.45	27.25, 0.47	17.64, 0.65	26.46, 0.44
DIA	**Population Size**	55	55	55	55	**Population Size**	55	55	55	55
	Maximum Generations	800	800	800	800	Δt , Z_{cr}	10.32, 0.43	17.33, 0.41	20.97, 0.51	18.73, 0.46
FEZ	**Population Size**	70	70	70	70	**Population Size**	70	70	70	70
	Maximum Generations	900	900	900	900	Δt , Z_{cr}	15.46, 0.52	19.74, 0.34	30.19, 0.35	19.12, 0.42
VGK	**Population Size**	50	50	50	50	**Population Size**	50	50	50	50
	Maximum Generations	850	850	850	850	Δt , Z_{cr}	20.78, 0.56	16.85, 0.62	10.37, 0.38	25.31, 0.34
EWG	**Population Size**	60	60	60	60	**Population Size**	60	60	60	60
	Maximum Generations	1000	1000	1000	1000	Δt , Z_{cr}	22.97, 0.48	32.43, 0.57	22.43, 0.31	27.23, 0.65
EEM	**Population Size**	65	65	65	65	**Population Size**	65	65	65	65
	Maximum Generations	900	900	900	900	Δt , Z_{cr}	30.45, 0.47	21.11, 0.71	31.17, 0.33	19.65, 0.68
ALL ETFs	**Selection Type**	Roulette Wheel Selection				**Foraging Speed**	0.02 ms^{-1}			
	Elitism	Best individual is kept in the next generation.				**Maximum Motion Speed**	0.01 ms^{-1}			
	Crossover Probability	0.9				**Maximum Diffusion Speed**	[0.002, 0.010] ms^{-1}			
	Mutation Probability	0.1				**Inertia Weights**	[0,1]			

Table B.2 presents the selected fitness functions for the proposed KH-LSVR$_1$ and KH-LSVR$_2$ models and their GA-εSVR and GA-vSVR benchmarks.

Table B.1: Selected Fitness Functions

	Series	GA-εSVR	GA-νSVR	KH-LSVR$_1$	KH-LSVR$_2$		Series	GA-εSVR	GA-νSVR	KH-LSVR$_1$	KH-LSVR$_2$
F1	SPY	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$	F2	SPY	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$
	QQQ	$Fitness_3$	$Fitness_3$	$Fitness_1$	$Fitness_3$		QQQ	$Fitness_1$	$Fitness_1$	$Fitness_3$	$Fitness_3$
	DIA	$Fitness_3$	$Fitness_2$	$Fitness_3$	$Fitness_1$		DIA	$Fitness_3$	$Fitness_2$	$Fitness_3$	$Fitness_2$
	FEZ	$Fitness_3$	$Fitness_3$	$Fitness_2$	$Fitness_3$		FEZ	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$
	VGK	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$		VGK	$Fitness_2$	$Fitness_3$	$Fitness_1$	$Fitness_3$
	EWG	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$		EWG	$Fitness_3$	$Fitness_3$	$Fitness_2$	$Fitness_3$
	EEM	$Fitness_2$	$Fitness_1$	$Fitness_3$	$Fitness_3$		EEM	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$
F3	SPY	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_1$	F4	SPY	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$
	QQQ	$Fitness_1$	$Fitness_2$	$Fitness_3$	$Fitness_3$		QQQ	$Fitness_3$	$Fitness_3$	$Fitness_1$	$Fitness_3$
	DIA	$Fitness_3$	$Fitness_3$	$Fitness_2$	$Fitness_3$		DIA	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$
	FEZ	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$		FEZ	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$
	VGK	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_1$		VGK	$Fitness_1$	$Fitness_1$	$Fitness_3$	$Fitness_3$
	EWG	$Fitness_2$	$Fitness_3$	$Fitness_1$	$Fitness_3$		EWG	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_2$
	EEM	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$		EEM	$Fitness_3$	$Fitness_3$	$Fitness_3$	$Fitness_3$

C. Statistical and trading performance measures.

The statistical and trading performance measures are calculated as shown in table C.1.

Table C.1: Statistical and Trading Performance Measures

STATISTICAL PERFOMANCE	DESCRIPTION
Root Mean Squared Error	$RMSE = \sqrt{\frac{1}{N'}\sum_{\tau=t+1}^{t+N'}(E(R_\tau)-Y_\tau)^2}$ *,with* Y_τ *being the actual value and* $E(R_\tau)$ *the forecasted value and* N' *the number of forecasts*
TRADING PERFOMANCE	**DESCRIPTION**
Annualized Return	$R^A = 252*\frac{1}{N'}*(\sum_{\tau=1}^{N'}R_\tau)-TC^A$ where R_τ is the daily return and TC^A is the annualized transaction cost
Information Ratio	$IR = \frac{R^A}{\sigma^A}$